IMPLEMENTASI *GRIPPER* PADA *END EFFECTOR* ROBOT UNTUK MEMEGANG TELUR AYAM DENGAN SENSOR *FSR (FORCE SENSITIVE RESISTOR)*

SKRIPSI

Untuk memenuhi sebagian persyaratan

memperoleh gelar Sarjana Teknik

Disusun oleh:

Alfian Reza Pahlevi

NIM: 145150300111053

PROGRAM STUDI TEKNIK KOMPUTER

JURUSAN TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2018

PENGESAHAN

IMPLEMENTASI *GRIPPER* PADA *END EFFECTOR* ROBOT UNTUK MEMEGANG TELUR AYAM DENGAN SENSOR *FSR (FORCE SENSITIVE RESISTOR)*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Alfian Reza Pahlevi
NIM: 145150300111053

Skripsi ini telah diuji dan dinyatakan lulus pada
2 Agustus 2018

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Dahnial Syauqy, S.T., M.T., M.Sc.	Bayu Rahayudi, S.T., M.T.
NIK: 201607 870423 1 002	NIP: 19740712 200604 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALTIAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia bahwa skripsi ini dibuktikan terdapat unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 1 Agustus 2018

Alfian Reza Pahlevi
NIM: 145150300111053

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Implementasi *Gripper* pada *End Effector Robot* untuk Memegang Telur Ayam dengan Sensor *FSR (Force Sensitive Resistor)*" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Allah SWT yang telah memberikan rahmat, taufik serta hidayah-Nya sehingga laporan skripsi dapat terselesaikan dengan baik.
2. Kedua orang tua dan seluruh keluarga besar atas segala doa, nasihat, kesabaran, dan dorongan semangat kepada penulis.
3. Bapak Wayan Firdaus Mahmudy, S.Si, M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Sabriansyah Rizqika Akbar, S.T., M.Eng. selaku Ketua Program Studi Teknik Komputer Fakultas Ilmu Komputer Universitas Brawijaya.
6. Bapak Dahnial Syauqy, S.T., M.T., M.Sc. selaku dosen pembimbing satu yang telah sabar memberikan nasihat dan ilmu dalam menyelesaikan skripsi ini.
7. Bapak Bayu Rahayudi, S.T., M.T. selaku dosen pembimbing dua yang telah sabar memberikan nasihat dan ilmu dalam menyelesaikan skripsi ini.
8. Seluruh dosen dan karyawan Fakultas Ilmu Komputer yang telah berperan dalam proses pengerjaan skripsi ini.
9. Teman-teman Teknik Komputer 2014 yang telah membantu memperlancar pengerjaan skripsi ini tidak dapat diucapkan satu persatu.
10. Teman-teman Fake Friends yang telah memberikan motivasi dan dukungan agar pengerjaan skripsi ini cepat terselesaikan.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata, semoga laporan skripsi penulis ini dapat bermanfaat bagi pembaca.

Malang, 1 Agustus 2018

Penulis
alfianrp12@gmail.com

ABSTRAK

Robot dengan *gripper* dapat membantu pengguna untuk mengambil objek di suatu tempat yang tak terjangkau manusia. *User* juga tidak perlu langsung berinteraksi dengan objek langsung karena robot dapat dikendalikan secara *nirkabel* dengan bantuan *wifi* dan perangkat Android. Sebagai pengganti indera perasa pada kulit manusia digunakanlah sensor *FSR (Force Sensitive Resistor).* Sensor ini digunakan untuk mendapatkan nilai keeratan genggaman *gripper* ketika mengambil objek. Untuk mengatasi kerusakan objek, *gripper* diimplementasikan dapat berhenti otomatis dengan menggunakan nilai *threshold*. *Threshold* yang dipakai adalah 552. Nilai 552 didapat dari hasil rerata pada 11 akuisisi data nilai sensor menggunakan sensor *FSR* dengan percobaan manual. Ketika nilai sensor yang didapat melewati *threshold*, maka gripper robot akan otomatis berhenti. Hal ini dilakukan untuk mencegah kerusakan pada objek telur ayam. Dari pengujian sebanyak 10 kali pada robot *gripper*, didapat hasil uji dengan persentase 100% bahwa sistem dapat menghentikan *gripper* dan objek telur ayam tidak rusak. Dari hasil uji didapat nilai tekanan yang berbeda-beda pada masing-masing objek. Hal ini bisa jadi disebabkan karena lapisan objek telur ayam yang beda atau nilai sensor yang terus berubah-ubah.

Kata kunci: Robot, *Gripper, FSR (Force Sensitive Resistor), WiFi,* Android*, Threshold*

ABSTRACT

Robots with gripper can help users to grab objects in a place that is unattainable to humans. Users also do not need to directly interact with the object directly because the robot can be controlled wirelessly with the help of wifi and Android devices. As a substitute for the senses of taste on human skin is used FSR sensor (Force Sensitive Resistor). This sensor is used to obtain the value of the gripper's closeness when grabbing the object. To overcome the damage of the object, the gripper is implemented can stop automatically by using the threshold value. The threshold used is 552. The value of 552 is obtained from the average result on the 11 data acquisition of sensor values using the FSR sensor with manual experiment. When the value of the sensor obtained through the threshold, then the robot gripper will automatically stop. This is done to prevent damage to the chicken egg object. From the test as much as 10 times on the robot gripper, obtained test results with 100% percentage that the system can stop gripper and chicken egg object is not damaged. From the test results obtained different pressure values on each object. This may be due to layers of chicken egg object is different or the value of the sensor that keeps changing.

Keywords: Robot, Gripper, FSR (Force Sensitive Resistor), WiFi, Android, Threshold

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Mobile robot dengan menggunakan *gripper* digunakan untuk mengambil objek yang sulit dijangkau atau objek yang berbahaya bagi manusia. Dalam hal ini pekerjaan manusia akan sangat terbantu dikarenakan manusia tidak langsung berinteraksi dengan objek yang dipegang. Maka dari itu dapat sistem *haptic* diimplementasikan untuk menggantikan rasa sentuhan jari manusia. Sistem haptic pada manusia digantikan oleh sensor *FSR* (*Force Sensitive Resistor*) yang diletakkan pada ujung *gripper robot*. Sensor *FSR* menggunakan nilai *threshold* yang bermacam-macam sebagai pengujian pada beberapa objek. Ketika nilai sensor melebihi *threshold*, maka perangkat Android pengguna akan bergetar (Zarkasih, et al., 2017).

Fungsi utama *gripper* adalah sebagai pemegang suatu benda. Pada penerapannya *gripper* dapat digunakan untuk *material handling* bermacam-macam barang hasil produksi manufaktur, contohnya: minuman, komponen mesin, mebel dan lain-lain. Prinsip kerja *gripper* dibantu dengan komponen lain seperti lengan penggerak *gripper*, poros penggerak *gripper* dan sistem *pneumatik* sebagai sumber energi untuk memegang suatu barang produksi (Isworo, et al., 2015).

Telur memiliki peran penting dalam pembuatan makanan seperti kue. Peran penting tersebut antara lain meningkatkan warna, mengatur struktur dan sebagai pengental. Jenis telur yang digunakan dalam formulasi kue juga dapat memengaruhi karakteristik kue seperti volume kue, tekstur serta stabilitas oksidatif kue. Telur ayam mempunyai beberapa potensi pemanfaatan dalam produksi kue karena dapat diperoleh dengan mudah. Kue yang dibuat dengan menggunakan telur ayam asli memiliki kandungan protein sedikit lebih tinggi daripada yang lain (Oyeyinka, et al., 2017).

Candling merupakan peneropongan telur dengan menggunakan lampu terang yang akan menunjukkan kecacatan seperti retakan, kotoran, dan bentuk telur yang tidak sesuai. Pada penelitian ini, menuliskan tentang otomatisasi dari *candling* dengan memakai robot dan kamera. Kamera video digunakan untuk mendeteksi telur yang retak dan lengan robot untuk mengambil telur tersebut. Setelah menemukan telur yang cacat, robot mengambilnya dengan lembut dan memindahkan telur cacat tersebut (Bourely, et al., 1987).

Dengan adanya *gripper*, maka telur ayam dapat dipegang dengan mudah oleh *robot*. Telur ayam memiliki sifat sensitif yang rentan pecah ketika dipegang tangan manusia karena adanya gesekan antar kulit manusia dengan cangkang telur ayam. Dengan adanya *gripper*, memegang telur akan lebih mudah dan mengurangi rasa sentuhan antar kulit manusia dengan cangkang telur ayam. Ada kelemahan ketika memegang telur ayam dilakukan oleh *gripper*. Ketika tidak ada batasan *gripper* untuk bergerak, maka akan menimbulkan resiko telur ayam pecah. Sebagai pengganti rasa sentuhan oleh jari manusia, digunakanlah sensor *FSR (Force Sensitive Resistor)*.

FSR (Force Sensitive Resistor) adalah alat yang digunakan untuk aplikasi atau komponen yang memerlukan analisis kekuatan. *FSR* lebih handal dan akurat dibanding *transduser* lainnya serta bisa menggambarkan karakteristik indera perasa. Pada kontrol tingkat tinggi, yang ada pada lengan *robot* adalah sistem cerdas untuk deteksi tekanan pada objek. Sistem umpan balik gaya akan menguntungkan sistem manipulasi objek karena resiko menjatuhkan atau menghancurkan objek akan berkurang dengan bantuan FSR (Kumar, et al., 2016).

Dalam mengoperasikan sebuah *robot* akan lebih mudah dilakukan dari jarak jauh. Apalagi posisi objek terletak pada tempat yang tidak bisa dijangkau oleh manusia. *Wifi* merupakan sebuah teknologi yang dapat menghubungakn suatu perangkat dengan perangkat lainnya melalui jaringan nirkabel seperti internet.

Oleh karena itu, dari semua permasalahan diatas, pada penelitian ini akan diimplementasikan *gripper* pada *end effector* robot untuk memegang telur ayam dengan menggunakan sensor *FSR* (*Force Sensitive Resistor*) yang dapat dikontrol secara nirkabel melalui perangkat Android. Pada pengaplikasiannya, perangkat Android akan memiliki beberapa tombol sebagai tombol operasional. Tombol pengontrol tersebut nantinya akan menimbulkan suatu perintah kepada *robot* secara *wireless* untuk menjalankan suatu tugas.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, permasalahan yang didapat dapat dirumuskan sebagai berikut:

1. Bagaimana cara mengetahui kekuatan *gripper* pada end *effector robot* untuk memegang telur ayam?
2. Bagaimana cara *gripper robot* berhenti ketika kekuatan tekanan sudah sesuai dengan yang diinginkan?
3. Bagaimana cara *robot* dapat menunjukkan nilai sensor dan dapat dikontrol secara nirkabel?

1.3 Tujuan

Berdasarkan latar belakang dan rumusan masalah yang ada, didapat tujuan penelitian ini adalah sebagai berikut:

1. Membuat *robot* dapat mengetahui seberapa erat kekuatan *gripper* pada end *effector* ketika memegang telur ayam.
2. Membuat *gripper* pada end *effector* robot dapat berhenti secara otomatis ketika sudah melewati batas keeratan ketika memegang telur ayam.
3. Membuat nilai sensor *FSR (Force Sensitive Resistor)* pada end *effector* robot dapat dimonitoring serta *gripper* dapat dikontrol secara nirkabel.

1.4 Manfaat

Berdasarkan latar belakang, rumusan masalah dan tujuan didapat manfaat yang didapat dari penelitian ini antara lain:

1. Membantu pengguna mendapatkan rasa sentuhan ketika sistem *gripper robot* menyentuh objek yang dipegang.
2. Membantu memberhentikan *gripper* ketika kekuatan keeratan sudah melebihi batas erat.
3. Menjadi gambaran untuk penelitian baru pada sistem *gripper* secara *nirkabel*.
4. Memudahkan pengguna untuk memegang telur ayam yang terletak di lokasi yang susah dijangkau oleh tangan manusia.

1.5 Batasan Masalah

Batasan masalah yang akan digunakan pada penelitian ini agar tidak terlalu meluas pembahasannya antara lain:

1. Penelitian ini berfokus pada objek telur ayam negeri yang berwarna kecoklatan.
2. Sensor yang digunakan untuk *gripper* pada end *effector robot* adalah sensor *FSR (Force Sensitive Resistor)*.
3. Nilai sensor yang digunakan dikonversi dari *analog* ke *digital*.
4. *Robot* memiliki 2 fungsi utama, yakni fungsi *gripper* dan fungsi naik turun lengan *robot*.
5. Robot dikendalikan secara nirkabel dengan bantuan *wifi*.

1.6 Sistematika Pembahasan

Pembahasan singkat dari masing-masing bab yang ada pada penelitian ini adalah sebagai berikut:

BAB 1 Pendahuluan

Bab ini berisi tentang latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, sistematika pembahasan dan jadwal pelaksanaan dari penelitian ini.

BAB 2 Landasan Kepustakaan

Bab ini menjelaskan tentang teori-teori yang terkait dengan penelitian ini serta penelitian-penelitian yang ada sebelumnya.

BAB 3 Metodologi Penelitian

Bab ini menjelaskan tentang langkah-langkah kerja yang dilakukan pada penelitian ini, seperti studi literatur, analisis kebutuhan, perancangan sistem dan implementasi sistem.

BAB 4 Rekayasa Kebutuhan

Bab ini menjelaskan tentang deskripsi umum sistem, kebutuhan perangkat keras dan lunak serta kebutuhan fungsional sistem.

BAB 5 Perancangan dan Implementasi

Bab ini menjelaskan tentang perancangan dan implementasi berupa perangkat keras dan perangkat lunak yang digunakan pada penelitian ini.

BAB 6 Pengujian dan Analisis

Bab ini menjelaskan tentang langkah kerja dalam melaksanakan pengujian sistem dan analisis pengujian yang dilakukan.

BAB 7 Penutup

Bab ini menjelaskan tentang kesimpulan dan saran yang didapat dari hasil hasil pengujian dan analisis.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini akan menjelaskan tentang tinjauan pustaka terhadap penelitian yang sudah ada sebelumnya dan dasar teori tentang perangkat keras serta perangkat lunak yang digunakan agar penelitian ini berjalan lebih baik.

2.1 Tinjauan Pustaka

Mobile robot dengan menggunakan *gripper* sebagai *end effector robot* digunakan untuk mengambil objek yang sulit dijangkau atau objek yang berbahaya bagi manusia. Dengan adanya *robot gripper* ini, peran manusia dapat digantikan dalam mengambil suatu objek. Sebagai pengganti rasa sentuh manusia, digunakanlah sistem *haptic* dengan menggunakan sensor *FSR* (*Force Sensitive Resistor*) yang diletakkan pada ujung *gripper robot*. Pada penelitiannya, digunakan berbagai macam nilai *threshold* sensor untuk mengukur suatu keeratan *gripper robot* terhadap objek. Ketika nilai sensor melebihi *threshold*, maka perangkat Android pengguna akan bergetar (Zarkasih, et al., 2017).

Gripper difungsikan sebagai pemegang suatu benda. Dalam penerapannya dapat digunakan untuk *material handling* bermacam-macam barang hasil produksi manufaktur seperti, contohnya: minuman, komponen mesin, mebel dan lain-lain. Prinsip kerja *gripper* dibantu dengan komponen lain seperti lengan penggerak *gripper*, poros penggerak *gripper* dan sistem pneumatik sebagai sumber energi untuk memegang suatu barang. Kegagalan dari sistem *material handling* dapat mengganggu proses produksi suatu barang, begitu juga *gripper* yang mengalami kegagalan dapat menurunkan jumlah produksi. Mekanisme kegagalan umumnya berasal dari salah operasi dan kelelahan sistem karena bekerja terus menerus untuk mengangkat beban. Sistem sudah bagus untuk masalah mengangkat beban, yakni dengan *gripper*. Namun, pada penelitian ini ditemukan suatu kejanggalan, yakni tidak adanya sensor *FSR* untuk merasakan apakah benda sudah terikat dengan erat atau belum (Isworo, et al., 2015).

Robot yang memakai dual *gripper* yang terdiri dari mesin *CNC* identik yang ditempatkan secara *linier* dan *material handling* untuk bongkar muat mesin dan mengangkut bagian-bagian antara keduanya. Dengan adanya dual *gripper*, peningkatan kinerja sistem dapat meningkat dan memberikan wawasan manajerial. Memang, jumlah *gripper* yang dipakai pada *robot* dapat memiliki jumlah besar dalam peningkatan kemampuannya. Jumlah *gripper* pada *robot* menentukan jumlah bagian yang bisa ditangani *robot* secara bersamaan. Namun,

untuk menambahkan tingkat tekanan genggaman perlu adanya sensor untuk mewakili indera perasa manusia (Gultekin, et al., 2017).

Berdasarkan penelitian-penelitian diatas, pada penelitian ini akan diimplementasikan *gripper* pada *end effector robot* untuk memegang telur ayam dengan menggunakan sensor *FSR* (*Force Sensitive Resistor*) yang dapat dikontrol melalui perangkat Android. Perangkat Android nantinya akan bisa memberikan perintah kepada *end effector robot* untuk mengeratkan *gripper* maupun untuk melonggarkannya. Dengan bantuan sensor *FSR* (*Force Sensitive Resistor*), suatu benda nantinya akan dapat terjapit dengan normal sehingga kerusakan suatu benda dapat diminimalkan. Semua sistem kontrol akan terkoneksi secara nirkabel melalui sambungan *wifi*.

2.2 Dasar Teori

2.2.1 Robot

Robot merupakan salah satu yang dapat digunakan untuk memenuhi tuntutan dalam membantu kebutuhan dari pekerjaan manusia. Salah satu dari jenis *robot* tersebut adalah *robot* lengan *(arm-robotic)* yang dapat dilihat seperti gambar 2.1. Umumnya *robot* lengan berfungsi untuk memindahkan suatu barang dari satu tempat ke tempat yang lain. Berbagai macam barang, mulai dari yang ringan sampai yang berat dan berbahaya. *Robot* tersebut dikendalikan oleh kontrol yang biasanya berupa tombol atau tuas, sehingga untuk mengoperasikan *robot* tersebut, sangat memerlukan keahlian khusus. Melihat kondisi itulah, maka diperlukan adanya *robot* lengan yang dapat dikendalikan tanpa menggunakan tombol atau tuas, akan tetapi juga dapat bergerak sesuai keinginan kita. *Robot* tersebut akan bergerak sesuai dengan perintah dari operator (Susa'at, 2015).

Gambar 2.1 Robot Lengan
Sumber: Susa'at (2015)

2.2.2 *Gripper* pada End *Effector Robot*

Gripper adalah sebagai pemegang suatu benda. Pada penerapannya *gripper* dapat digunakan untuk *material handling* bermacam-macam barang hasil produksi manufaktur, contohnya: minuman, komponen mesin, mebel dan lain-lain. Prinsip kerja *gripper* dibantu dengan komponen lain seperti lengan penggerak *gripper*, poros penggerak *gripper* dan sistem pneumatik sebagai sumber energy untuk memegang suatu barang produksi (Isworo, et al., 2015). Contoh *gripper* pada *end effector* robot dapat dilihat seperti gambar 2.2.

Gambar 2.2 *Gripper* Robot
Sumber: Fuster (2015)

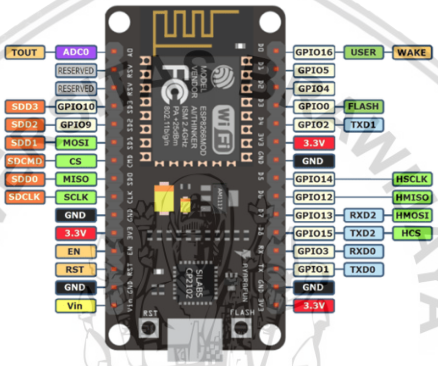
2.2.3 Telur Ayam

Ada beberapa jenis telur yang biasa dikonsumsi yaitu telur ayam, itik, angsa, burung puyuh, dan jenis telur unggas lainnya mempunyai struktur yang sama. Bagian yang terbesar dari telur adalah terdiri dari bahan organik, yaitu protein, lemak, karbohidrat dan garam mineral. Jenis telur ayam negeri biasa dikonsumsi oleh masyarakat, meskipun banyak jenis telur lain seperti telur bebek, telur angsa, atau telur burung puyuh yang juga dapat dikonsumsi. Berat telur ayam negeri sekitar 40-50 g dan lebar telur sekitar 4 cm serta warna cangkang cokelat gelap hingga terang (Boga, 2006). Contoh telur ayam negeri seperti gambar 2.3.

Gambar 2.3 Telur Ayam Negeri
Sumber: Pratama (2017)

2.2.4 Mikrokontroller NodeMCU

NodeMCU ESP8266 merupakan modul turunan pengembangan dari modul platform *IoT (Internet of Things)* keluarga ESP8266 tipe ESP-12. Secara fungsi modul ini hampir menyerupai dengan platform modul arduino, tetapi yang membedakan yaitu dikhususkan untuk *"Connected to Internet".* Versi 1.0 merupakan pengembangan dari versi 0.9. Versi 1.0 ini ESP8266 yang digunakan yaitu tipe ESP-12E yang dianggap lebih stabil dari ESP-12. Selain itu ukuran board modulnya diperkecil sehingga compatible digunakan membuat prototipe projek di breadboard. Serta terdapat pin yang dikhusukan untuk komunikasi *SPI (Serial Peripheral Interface)* dan *PWM (Pulse Width Modulation)* yang tidak tersedia di versi 0.9. Untuk detail pin lebih jelas dapat dilihat pada gambar 2.4.

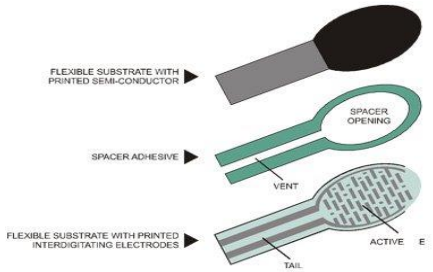


Gambar 2.4 Pin Out NodeMCU Versi 1.0

Sumber: Admin (2017)

2.2.5 Sensor *FSR (Force Sensitive Resistor)*

FSR adalah sensor yang mendeteksi tekanan fisik. Sensor *FSR* mudah digunakan dan harganya murah. Sensor *FSR* seperti pada gambar 2.5 terbuat dari 2 lapisan yang dipisahkan oleh spacer. Semakin banyak penekanan, semakin banyak titik elemen aktif yang menyentuh semikonduktor itu dan membuat resistansi menurun. Bila tidak ada tekanan, maka sensor terlihat seperti *resistor* tak terhingga, seiring dengan meningkatnya tekanan, resistansi turun.



Gambar 2.5 Sensor *FSR* (*Force Sensitive Resistor*)

Sumber: Ladyada (2013)

Untuk mendapatkan nilai sensor akan dibutuhkan resistor 10K Ohm dengan memanfaatkan rangkaian pembagi tegangan. Tegangan yang masuk ke sensor sebesar 3.3v sesuai dengan tegangan pada mikrokontroller NodeMCU. Sehingga untuk mendapatkan nilai tegangan keluaran dapat menggunakan rumus (2.1).

$$Vout = Vin \times \frac{R1}{R1+R.FSR} \text{ .....(2.1)}$$

Keterangan:

V_{out} : Tegangan yang dihasilkan oleh pembagi tegangan

V_{in} : Tegangan yang masuk ke sensor FSR sebesar 3.3v

R_1 : Resistor 10k Ohm

R_{FSR} : Resistansi sensor FSR

2.2.6 *ADC (Analog to Digital Converter)*

ADC (Analog to Digital Converter) fitur yang sangat berguna untuk mengubah tegangan analog menjadi digital. Dengan perubahan dari analog ke digital, sensor dapat dengan mudah dibaca oleh pin digital pada mikrokontroller. Cara kerja ADC cukup kompleks. Ada salah satu teknik yang paling umum digunakan, yakni dengan menggunakan tegangan analog untuk mengisi sebuah kapasitor internal dan kemudian mengukur waktu yang dibutuhkan untuk melepaskan *resistor* internal. Untuk rumus sederhana bisa memakai rumus (2.2).

$$\frac{Resolution\ of\ the\ ADC}{System\ Voltage} = \frac{ADC\ Reading}{Analog\ Voltage\ Measured} \text{ .....(2.2)}$$

Resolution adalah jumlah bit pada pin digital. Misal 10bit, jadi 1023. System voltage adalah voltase maks pada pin digital itu sendiri, misal 5v. Analog voltase adalah voltase yang didapat dari analog, misal 3v. Jadi, nanti akan menghasilkan ADC sebesar 614. Contoh perhitungan rumus (2.2) dapat dilihat dibawah ini.

$$\frac{1023}{3} = \frac{ADC\ Reading}{2}$$

$$ADC = \frac{1023 * 2}{3}$$

$$ADC = \frac{2046}{3}$$

$$ADC = 682$$

2.2.7 Motor *Servo*

Motor *servo* adalah sebuah motor DC yang dilengkapi rangkaian kendali dengan sistem *closed feedback* yang terintegrasi dalam motor tersebut. Pada

motor *servo* posisi putaran sumbu (*axis*) dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor servo seperti pada gambar 2.6 disusun dari sebuah motor DC, *gearbox*, *variabel resistor* (VR) atau *potensiometer* dan rangkaian kontrol. *Potensiometer* berfungsi untuk menentukan batas maksimum putaran sumbu (*axis*) motor *servo*. Sedangkan sudut dari sumbu motor *servo* diatur berdasarkan lebar pulsa atau biasa disebut *PWM (Pulse Width Modulation)*. Motor *servo* akan mnejaga suatu posisi ketika mendapat perintah untuk menuju ke posisi tersebut (Purnama, 2012).

Gambar 2.6 Motor Servo
Sumber: Purnama (2012)

2.2.8 LCD

LCD *(Liquid Crystal Display)* adalah komponen elektronika yang mempunyai fungsi untuk menampilkan suatu karakter baik angka maupun huruf. Pada modul LCD seperti gambar 2.7 terdapat *mikrokontroller* yang berfungsi sebagai pengendali tampilan karakter. *Mikrokontroller* pada suatu LCD dilengkapi dengan memori dan *register*. Pin input pada LCD antara lain pin RS *(Register Select)*, pin E *(Enable)*, pin VLCD, pin data, dan pin R/W *(Read Write)*. Format dan jumlah karakter yang dapat ditampilkan oleh LCD berbeda tiap jenis dan ukurannya. Untuk ukuran 16x2 artinya LCD terdapat 2 baris dan 16 kolom tiap barisnya. (Purnama, 2012).

Gambar 2.7 *LCD* 16x2
Sumber: Purnama (2012)

2.2.9 I2C

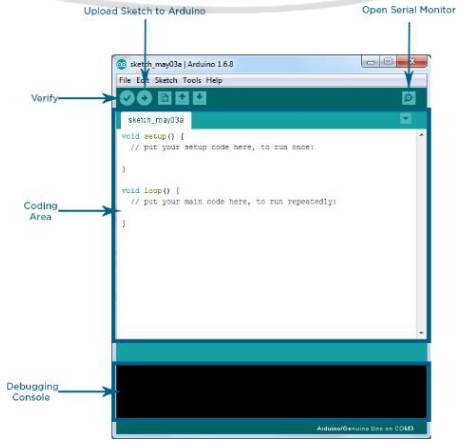
Adapter Serial I2C LCD seperti gambar 2.8 mempunyai fungsi untuk mengkonversi 16 x 2 karakter layar LCD menjadi LCD seri i2C yang dapat dikontrol melalui hanya 2 kabel, SDA dan SCL. Adapter yang digunakan adalah *chip* PCF8574 yang berfungsi sebagai I / O *expander* untuk berkomunikasi dengan Arduino atau *mikrokontroler* lainnya. I2C mempunyai alamat setiap terhubung dengan LCD dan mikrokontroller melalui kabel SDA dan SCL. Alamat i2C *default* adalah 0X27 dan dapat berubah tergantung sambungan pada pin *mikrokontroller*. Tegangan yang digunakan juga umum, yakni 5v. Terdapat *potensiometer* pada adapter I2C sebagai pengontrol kontras LCD.

Gambar 2.8 *Adapter Serial* I2C

Sumber: Tomi (2014)

2.2.10 Arduino *IDE*

Arduino adalah platform elektronik *open source* yang berbasis pada perangkat keras dan perangkat lunak yang mudah digunakan. *Board* Arduino dapat membaca masukan seperti sensor, jari pada tombol dan mengubahnya menjadi output seperti mengaktifkan motor, menyalakan LED, mempublikasikan sesuatu secara online. User bisa memberi intruksi kepada *Arduino IDE* tentang apa yang harus dilakukan dengan mengirimkan satu set instruksi ke mikrokontroler di *board Arduino*. Untuk memulainya, dapat dilihat halaman antarmuka pemrograman seperti pada gambar 2.9.



Gambar 2.9 Tampilan Arduino IDE

Sumber: Aidan (2016)

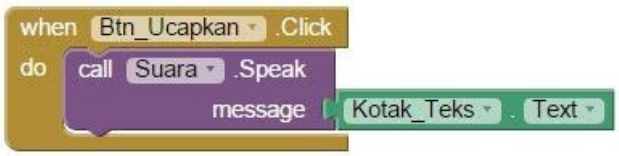
2.2.11 MIT App Inventor

MIT App Inventor adalah bentuk pemrograman visual yang intuitif dan membuatnya bisa dipakai semua kalangan bahkan anak-anak untuk membuat aplikasi Android. Bentuk antarmuka pemrograman MIT App Inventor dapat dilihat pada gambar 2.10. MIT App Inventor menyediakan blok-blok program yang kompleks dalam pembuatan aplikasi Android dan hanya membutuhkan waktu yang relatif sedikit daripada pemrograman tradisional. Proyek MIT App Inventor berusaha untuk memberdayakan semua orang agar mudah membuat aplikasi Android, terutama pemuda untuk beralih dari penikmat teknologi menjadi pencipta teknologi baru. MIT App Inventor mempunyai fitur yang banyak sehingga pembuat dapat leluasa membuat aplikasi Android (Abelson, H., 2017).

Gambar 2.10 Tampilan MIT App Inventor

Sumber: Abelson (2017)

Selain membuat antarmuka aplikasi Android, MIT App Inventor juga mengatur *event* pada fungsi tampilan yang telah dibuat. *Event* ini berguna agar tiap komponen dapat berfungsi sesuai keinginan. Untuk membuat *event* ini dapat diatur pada *Blocks* program. Sebuah block dapat dianalogikan sebagai potongan kata atau kalimat, yang mana apabila kata tersebut disusun dengan benar dia akan membentuk sebuah kombinasi kalimat yang memberikan instruksi kepada handphone anda untuk melakukan sesuatu. Block juga memiliki bentuk dan warna tertentu seperti pada contoh gambar 2.11 (Amiroh, 2015).



Gambar 2.11 *Block* Program

Sumber: Amiroh (2015)

2.2.12 WiFi

Wi-Fi merupakan singkatan dari *Wireless Fidelity* yaitu sebuah media penghantar komunikasi data tanpa kabel yang bisa digunakan untuk komunikasi atau mentransfer program dan data. Wi-Fi juga dapat diartikan teknologi yang memanfaatkan peralatan elektronik untuk bertukar data dengan menggunakan gelombang radio (nirkabel) melalui sebuah jaringan komputer termasuk koneksi berkecepatan tinggi. Untuk penggunaan wifi memerlukan titik akses yang disebut dengan hotspot untuk mengontrol antar pengguna *wifi* (Karim, 2016).

2.2.13 HTTP

Hypertext Transfer Protokol (HTTP) merupakan protokol yang menyediakan perintah dalam komunikasi antar jaringan, yaitu komunikasi antara jaringan komputer *client* dengan *web server*. Dalam komunikasi ini, komputer *client* melakukan permintaan dengan mengetikkan alamat atau *website* yang ingin di akses. Sedangkan *server* mengolah permintaan tersebut berdasarkan kode protokol yang di inputkan. Sebuah sesi HTTP terdiri dari *request* dan *respons*. Sebuah *client* HTTP akan memulai sebuah permintaan dengan membuka sebuah koneksi ke sebuah port 80. Server yang mendengarkan pada port 80 tersebut akan menunggu pesan permintaan *client*. Saat menerima permintaan, server akan mengirimkan kembali baris status, seperti "HTTP/1.1 200 OK" atau informasi lainnya (Astriani, 2013).

2.2.14 *Library* ESP8266WiFi.h

Library ini berfungsi untuk mengaktifkan fungsi koneksi *wifi* yang ada pada mikrokontroller NodeMCu. Koneksi *wifi* yang digunakan disambungkan melalui *Network SSID* dan password sebagai penghubung dengan perangkat Android. *Network SSID* dan password digunakan agar dapat dihubungkan dengan beberapa perangkat saja, artinya privasi koneksi terjaga.

2.2.15 *Library* Servo.h

Library ini digunakan untuk mengontrol motor servo yang terhubung dengan mikrokontroller. Pin yang digunakan adalah pin digital mikrokontroller. Untuk mengatur servo ini cukup dengan fungsi *write* yang sudah ada pada *library*. Yang dimasukkan pada fungsi *wirte* cukup angka derajat dan *servo* akan menjaga kondisi pada derajat tersebut.

2.2.16 *Library* Wire.h

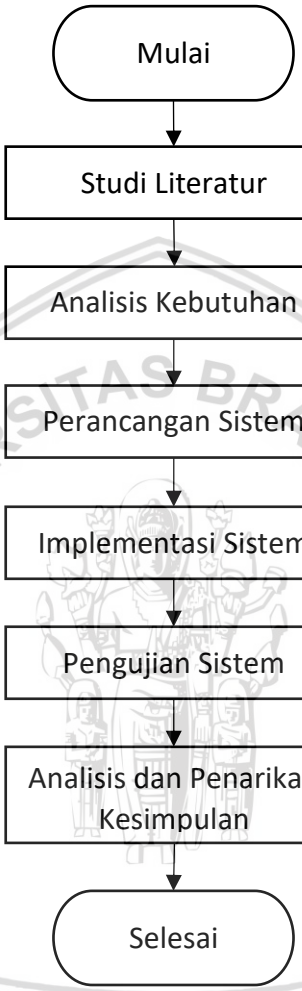
Library ini digunakan untuk menangani protokol serial sinkron secara I2C. Beberapa fugsi yang terdapat pada *library* ini antara lain begin() untuk inisialisasi protokol I2C, write() untuk mengirim data serial, dan read() untuk membaca data srial. Pada skripsi ini *library* ini digunakan untuk menampilkan karakter pada LCD.

2.2.17 *Library* LiquidCrystal_I2C.h

Library ini digunakan untuk menampilkan karakter pada LCD. *Library* ini digunakan untuk LCD yang sudah tersambung dengan I2C. Dengan *library* ini, 2 pin I2C yakni SDA dan SCL dapat berfungsi dengan baik.

BAB 3 METODOLOGI

Pada bab ini menjelaskan tentang langkah-langkah metodologi dalam penulisan skripsi. Diagram alir metodologi dapat dilihat pada gambar 3.1.



Gambar 3.1 Diagram Alir Metodologi

3.1 Studi Literatur

Studi literatur untuk menjelaskan tentang dasar teori yang digunakan dalam menyelesaikan masalah pada perancangan dan implementasi sistem. Literatur didapat dari buku, jurnal, website maupun penelitian yang pernah ada sebelumnya. Berikut hal-hal yang menjadi studi literatur:

1. Robot
2. *Gripper* pada End *Effector*
3. Telur Ayam
4. Mikrokontroller NodeMCU

5. Sensor *FSR (Force Sensitive Resistor)*
6. ADC *(Analog to Digital Converter)*
7. Motor Servo
8. LCD
9. I2C
10. Arduino IDE
11. MIT App Inventor
12. WiFi
13. HTTP

3.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan gambaran umum tentang semua kebutuhan yang diperlukan selama merancang dan membangun sistem. Analisis kebutuhan yang diperlukan meliputi analisis kebutuhan perangkat keras, analisis kebutuhan perangkat lunak dan analisis kebutuhan fungsional. Adapun kebutuhan akan dijelaskan sebagai berikut:

3.2.1 Analisis Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras dilakukan untuk mengetahui perangkat keras apa saja yang digunakan beserta fungsi. Adapun perangkat keras yang dibutuhkan antara lain Mikrokontroller NodeMcu, Sensor *FSR*, *Gripper*, Motor Servo, LCD, I2C dan Perangkat Android. Untuk penggunaan masing-masing perangkat keras akan dijelaskan lebih detail pada bab selanjutnya.

3.2.2 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak dilakukan untuk mengetahui perangkat lunak apa saja yang digunakan beserta fungsi. Adapun perangkat lunak yang dibutuhkan adalah Arduino IDE dan MIT App Inventor. Untuk penggunaan masing-masing perangkat lunak akan dijelaskan lebih detail pada bab selanjutnya.

3.2.3 Kebutuhan Fungsional

Kebutuhan fungsional dilakukan untuk mengetahui apa saja fungsi yang harus dicapai dari penelitian ini. Adapun kebutuhan fungsional yang terdapat pada penelitian ini salah satunya adalah sistem dapat mengendalikan *gripper* pada end *effector robot* dengan fungsi mengeratkan dan melonggarkan. Untuk penjelasan lebih detail akan dijelaskan pada bab selanjutnya.

3.3 Perancangan Sistem

Perancangan sistem bertujuan untuk mendapatkan gambaran umum tentang semua kebutuhan yang diperlukan selama merancang dan membangun sistem. Adapun diagram blok yang diperlukan untuk membangun sistem dijelaskan lebih rinci pada bab 5.1.

3.4 Implementasi Sistem

Implementasi sistem bertujuan untuk membentuk semua komponen agar terlihat rapi ketika dilakukan pengujian. Sebelum ke tahap implementasi, terlebih dahulu dilakukan pengecekan pada masing-masing komponen yang digunakan dalam kondisi baik. Selanjutnya, komponen-komponen seperti sensor *FSR*, motor servo dan LCD I2C dihubungkan ke NodeMCU. Lalu motor servo disambungkan ke *gripper* yang difungsikan untuk mengeratkan atau melonggarkan. Lalu, konfiguasi *wifi* agar robot dapat terkoneksi dengan perangkat *Android*.

3.5 Pengujian Sistem

Pengujian sistem dilakukan berdasarkan implementasi sistem yang sudah dilakukan. Adapun parameter-parameter yang diujikan pada sistem antara lain:

1. Pengujian mengontrol robot secara *nirkabel* dengan menggunakan perintah dari perangkat Android.
2. Pengujian akuisisi data sensor ketika sensor melekat pada telur ayam.
3. Pengujian penghentian gripper secara otomatis ketika threshold nilai sensor telah melewati batas.
4. Pengujian keeratan pada telur ayam ditinjau dari nilai sensor.

3.6 Analisis dan Penarikan Kesimpulan

Analisis akan dilakukan setelah pengujian pada sistem. Kesimpulan merupakan jawaban dari rumusan masalah yang telah dibahas sebelumnya. Dari hasil analisis dan penarikan kesimpulan nantinya akan memunculkan saran untuk penelitian selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

Pada bab ini akan dijelaskan tentang berbagai kebutuhan yang dibutuhkan oleh sistem, dimulai dari gambaran umum sistem, kebutuhan sistem, batasan perancangan, serta asumsi dan ketergantungan dari sistem.

4.1 Gambaran Umum Sistem

Sistem yang dibangun secara umum berfungsi untuk memegang telur ayam dengan menggunakan *gripper robot*. Kekuatan pegangan didapat dari sensor *FSR (Force Sensitive Resistor)* yang dipasang pada ujung *gripper*. Nilai sensor diubah terlebih dahulu dari analog ke digital. Semakin besar nilai digital maka akan semakin besar pula keeratannya. Keeratan pegangan *gripper* mempunyai batas yang disebut *threshold*. Ketika sensor mencapai *threshold*, maka *gripper* akan berhenti menjapit secara otomatis. Sistem juga dilengkapi dengan alat gerak lengan robot untuk mengarahkan *gripper* pada titik letak yang pas untuk menjapit.

4.2 Kebutuhan Sistem

Pada kebutuhan sistem akan menjelaskan tentang apa saja yang dibutuhkan sistem untuk berjalan sebagaimana mestinya, mulai dari kebutuhan fungsional, kebutuhan perangkat lunak dan kebutuhan perangkat keras.

4.2.1 Kebutuhan Fungsional

Kebutuhan fungsional adalah kebutuhan untuk mengetahui apa saja fungsi yang harus dicapai dari penelitian ini. Adapun kebutuhan fungsional yang terdapat pada penelitian ini adalah sebagai berikut:

1. **Sistem dapat mengendalikan *gripper* pada *end effector* robot dengan fungsi mengeratkan dan melonggarkan.**

   Sistem akan dapat mengeratkan dan melonggarkan *gripper* dari perintah yang ada pada perangkat Android. Perintah yang ada yakni dari tombol “>> <<” untuk mengeratkan dan tombol “<< >>” untuk melonggarkan.

2. **Sistem dapat mengakuisisi data keeratan cengkraman pada *gripper.***

   Sistem akan dapat mendapatkan data keeratan pada cengkraman *gripper* melalui data sensor. Nilai sensor nantinya akan diubah dari *analog* ke *digital*. Nilai ADC berasal dari nilai pembagi tegangan pada rangkaian sensor ke sistem. Nantinya, nilai sensor ini akan ditampilkan pada LCD agar bisa dilihat pada keeratan berapa sensor sudah melewati *threshold*.

3. **Sistem dapat menghentikan *gripper* ketika *threshold* mencapai batas tertentu.**

   Nilai sensor yang sudah diubah dari analog ke digital akan dijadikan sebagai parameter untuk menghentikan *gripper robot*. Pada saat nilai sensor melebihi nilai *threshold* sensor, maka *gripper robot* akan secara otomatis berhenti bergerak. Nilai *threshold* sendiri sudah didefinisikan pada sistem.

4. **Sistem dapat menampilkan informasi pada LCD.**

   Sistem akan dipasang LCD I2C untuk menampilkan informasi pada *mikrokontroller*. Beberapa informasi penting tersebut antara lain *IP Address NodeMCU* dan nilai sensor. *IP Address NodeMCU* digunakan sebagai identittas sistem untuk disambungkan dengan perangkat Android secara *nirkabel*. Sedangkan, nilai sensor digunakan sebagai acuan nilai keeratan *gripper* ketika sudah melewati *threshold*.

4.2.2 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras adalah kebutuhan untuk mengetahui perangkat keras apa saja yang digunakan pada sistem. Adapun perangkat keras yang dibutuhkan adalah sebagai berikut:

1. **Mikrokontroller *NodeMCU***

   Mikrokontroller *NodeMCU* digunakan sebagai modul pemrosesan yang memiliki fitur *wifi* sehingga suatu sistem dapat tersambung dengan perangkat Android.

2. **Sensor *FSR (Force Sensitive Resistor)***

   Sensor *FSR (Force Sensitive Resistor)* digunakan sebagai input dari nilai keeratan *gripper* ketika menyentuh objek.

3. ***Gripper***

   Gripper digunakan sebagai alat *fleksible* buka tutup dengan ujung yang berfungsi sebagai pemegang objek.

4. **Motor *servo***

   Motor *servo* digunakan sebagai penggerak *gripper* dan lengan robot.

5. **LCD**

   LCD digunakan untuk menampilkan informasi dari sistem.

6. **I2C**

   I2C digunakan untuk meminimalisir penggunaan pin LCD menjadi 4 pin.

7. **Perangkat Android**

   Perangkat Android digunakan sebagai pengontrol sistem.

4.2.3 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak adalah kebutuhan untuk mengetahui perangkat lunak apa saja yang digunakan pada sistem. Adapun perangkat lunak yang dibutuhkan adalah sebagai berikut:

1. **Arduino IDE**

   Arduino IDE digunakan untuk memprogram mikrokontroller *NodeMCU* agar bekerja sebagaimana mestinya.

2. **Android Studio**

   Android Studio digunakan untuk membuat aplikasi pada perangkat Android serta antarmuka pengguna.

3. ***Library* ESP8266WiFi.h**

   Library ESP8266WiFi.h digunakan untuk koneksi *wifi* pada *NodeMCU.*

4. ***Library* Servo.h**

   Library Servo.h digunakan untuk menggerakkan motor *servo.*

5. ***Library* Wire.h**

   Library Wire.h digunakan untuk komunikasi serial pada adapter I2C.

6. ***Library* LiquidCrystal_I2C.h**

   Library LiquidCrystal_I2C.h digunakan untuk menampilkan karakter di LCD.

4.3 Batasan Perancangan Sistem

Pada tahap perancangan sistem akan membahas tentang batasan yang digunakan agar penelitian lebih terarah sesuai tujuan. Adapun batasan perancangan sistem adalah sebagai berikut:

1. Objek yang dipakai hanya telur ayam negeri warna kecokelatan.
2. *Gripper* akan terbuka penuh dengan lebar 4.8 cm.
3. *Gripper* akan berhenti ketika nilai sensor melewati *threshold*.
4. Objek yang dipegang langsung menyentuh sensor *FSR*.

5. Objek telur ayam yang dipegang dalam posisi tidur, bukan berdiri.

4.4 Asumsi dan Ketergantungan

Asumsi dan ketergantungan adalah dugaan sistem yang akan beroperasi nantinya. Adapun asumsi dan ketergantungan adalah sebagai berikut:

1. Objek dan sistem harus pada satu bidang datar.
2. Objek yang dipegang berada diantara ujung *gripper*.
3. *Gripper* menyentuh tepat ditengah kulit objek.
4. *Gripper* akan bergerak menutup ketika sudah dikontrol melalui perangkat.
5. Sistem berjalan ketika sudah tersambung dengan *wifi*.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

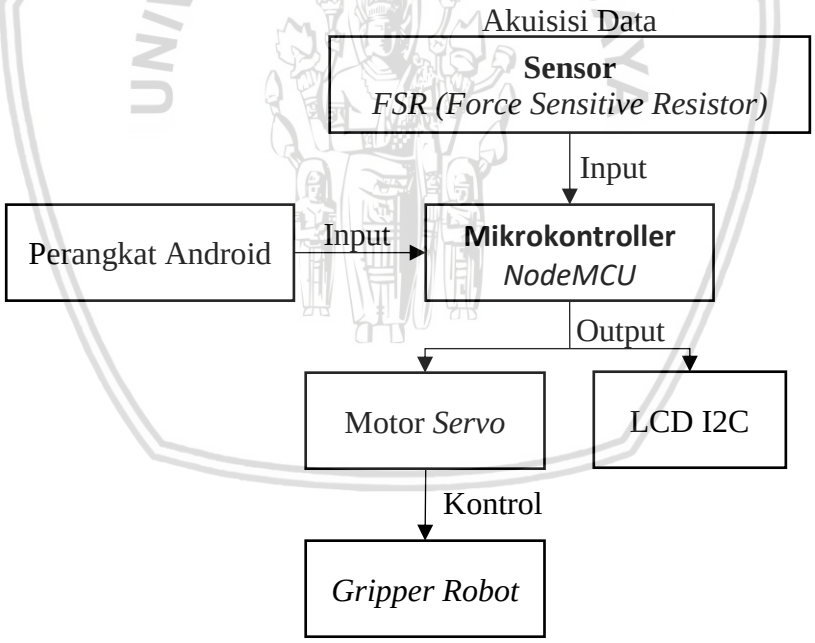
Pada bab ini akan dijelaskan tentang perancangan sistem dan implementasi sistem agar dapat berjalan dengan baik.

5.1 Perancangan Sistem

Pada tahap ini akan dijelaskan mengenai perancangan perangkat keras, perangkat lunak dan penentuan nilai *threshold* pada sistem dan aplikasi Android.

5.1.1 Perancangan Perangkat Keras

Pada tahap ini akan dijelaskan mengenai perancangan perangkat keras pada sistem. Perancangan perangkat keras terdiri dari perancangan rangkaian elektronik, letak posisi *gripper,* dan letak posisi sensor. Perancangan perangkat keras akan dijelaskan lebih detail pada setiap komponennya. Mulai dari posisi komponen sampai pin-pin yang dipakai.



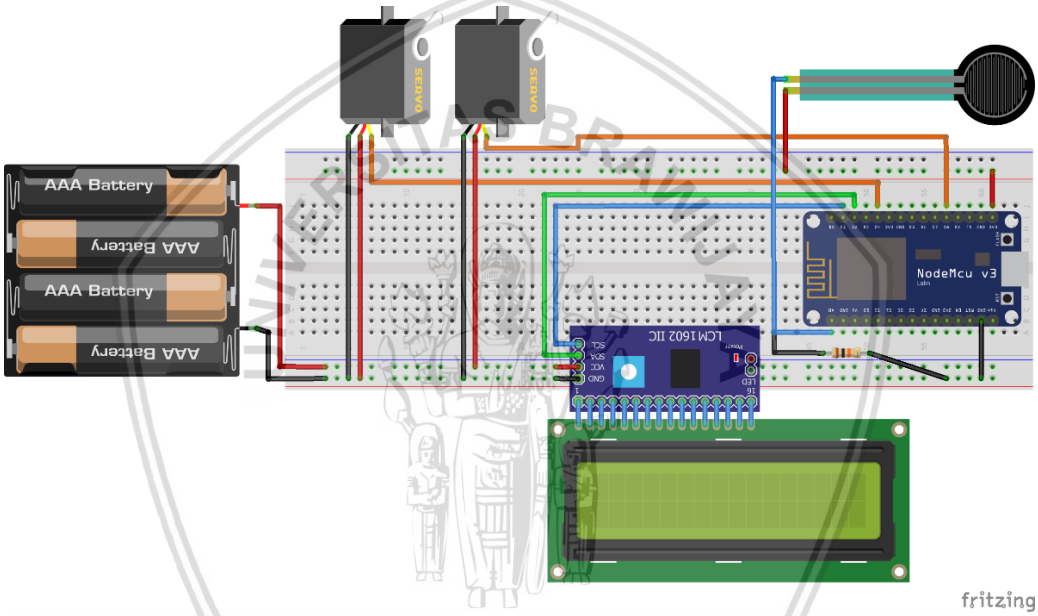
Gambar 5.1 Block Diagram Sistem

Pada gambar 5.1 menunjukkan semua perangkat keras yang dipakai sistem. Sensor FSR akan dijadikan input untuk mendapatkan nilai keeratan. *Mikrokontroller* NodeMCU akan dijadikan sebagai pemroses. Perangkat Android akan dijadikan sebagai pengontrol *servo*. Motor *servo* akan dijadikan sebagai penggerak *gripper*. LCD I2C akan dijadikan sebagai penampil informasi yang ada pada mikrokontroller.

5.1.1.1 Perancangan Rangkaian Elektronik

Pada tahap ini akan dijelaskan mengenai rangkaian sistem yang menyambungkan antar komponen yang diperlukan. Komponen yang disusun antara lain:

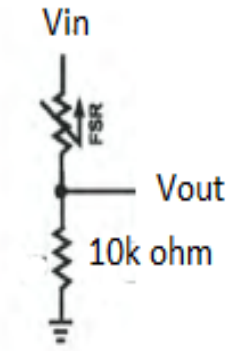
1. Sensor *FSR (Force Sensitive Resistor)*
2. Mikrokontroller *NodeMCU*
3. Motor Servo
4. LCD
5. I2C
6. Power Supply 5v



Gambar 5.2 Skema Rangkaian Sistem

Gambar 5.2 merupakan skema gambar rangkaian yang akan diimplmentasikan pada sistem. Mulai dari tegangan yang digunakan disambungkan ke mikrokontroller *NodeMCU*, *LCD I2C* dan motor *servo*. Untuk membaca sensor dibutuhkan pin *analog* yang terdapat pada mikrokontroller *NodeMCU*. Motor *servo* disambungkan pada pin digital mikrokontroller *NodeMCU*. Lalu *LCD I2C* disambungkan pada pin *digital mikrokontroller NodeMCU*.

R_1 merupakan pembagi tegangan seperti pada gambar 5.3. Ketika terdapat tekanan pada FSR nilai resistansi FSR akan menurun sehingga nilai V_{out} akan bertambah, begitu pula sebaliknya. Jika tekanan hilang makan V_{out} akan menurun. Nilai V_{out} akan masuk ke A0 *NodeMCU* seperti pada gambar 5.4.



Gambar 5.3 Rangkaian Pembagi Tegangan

Pada perancangan sensor FSR didapat bahwa sensor mempunyai $_{Vout}$ minimal dan maksimal ketika sensor mengenai objek. V_{out} minimal sensor ketika resistansi sensor sebesar 100k ohm, sedangkan V_{out} maksimal sensor ketika resistansi sensor sebesar 200 ohm. Jika ditinjau dari rumus perhitungan V_{out} 2.1, didapat hasil bahwa V_{out} minimal sebesar 0,3v dan V_{out} maksimal sebesar 3,24v. Perhitungan lebih detail seperti berikut.

$$Vout = 3{,}3\ x\ \frac{10K}{10K + 100K}$$

$$Vout = 3{,}3\ x\ \frac{10K}{110K}$$

$$Vout = 3{,}3\ x\ 0{,}09$$

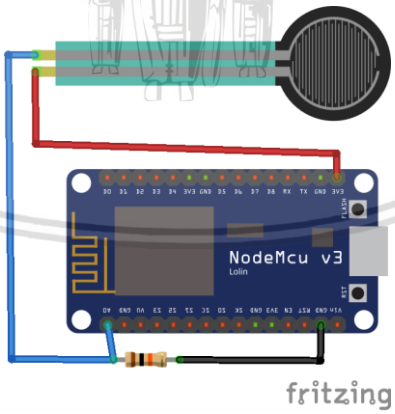
$$Vout = 0{,}3v$$

$$Vout = 3{,}3\ x\ \frac{10000}{10000 + 200}$$

$$Vout = 3{,}3\ x\ \frac{10000}{10200}$$

$$Vout = 3{,}3\ x\ 0{,}98$$

$$Vout = 3{,}24v$$



Gambar 5.4 Rangkaian Sensor FSR dengan NodeMCU

Pada perancangan sensor dengan NodeMCU dapat dilihat bahwa ada konversi nilai dari *analog* ke *digital* sehingga nilai sensor dapat diterima oleh mikrokontroller. Nilai ADC minimal ketika V_{out} 0,3v sedangkan nilai ADC maksimal ketika V_{out} 3,24v. Jika ditinjau dari rumus perhitungan ADC 2.2, didapat hasil bahwa nilai ADC yang terbaca antara 93 sampai 1060 dari data 10bit yang bisa diterima oleh NodeMCU. Perhitungan lebih detail seperti berikut.

$$\frac{1023}{3,3} = \frac{ADC\ Reading}{0,3}$$

$$\frac{1023\ x\ 0,3}{3,3} = ADC\ Reading$$

$$ADC\ Reading = \frac{306,9}{3,3}$$

$$ADC\ Reading = 93$$

$$\frac{1023}{3,3} = \frac{ADC\ Reading}{3,42}$$

$$\frac{1023\ x\ 3,42}{3,3} = ADC\ Reading$$

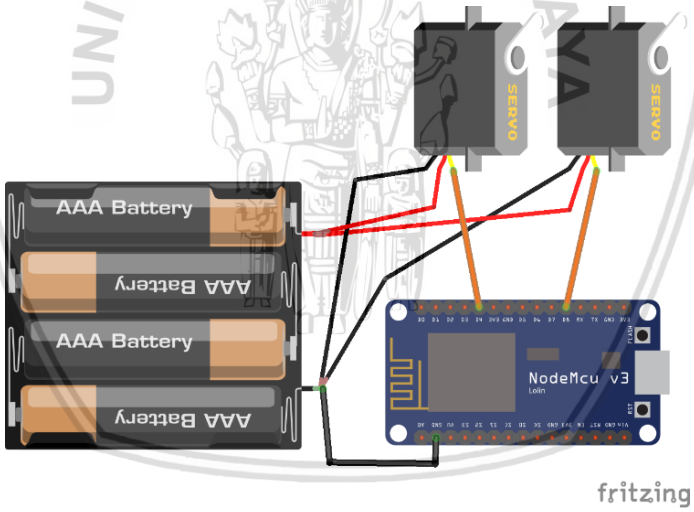
$$ADC\ Reading = \frac{3498,66}{3,3}$$

$$ADC\ Reading = 1060$$

Tabel 5.1 Koneksi Pin Motor Servo dengan *NodeMCU*

Motor Servo	*NodeMCU*	Fungsi	Power 5v
DATA	D4, D8	Data	
VCC		Power Supply	+
GND	G	Ground	-

Pada tabel 5.1 dapat dilihat bahwa pin VCC pada *servo* dihubungkan pada tegangan 5v yang dihasilkan dari adaptor. Pin *GND* dihubungkan pada pin G *NodeMCU* dan *ground* adaptor. Pin DATA pada *servo* dihubungkan ke pin *digital* D4 dan D8 pada NodeMCU. D4 untuk *servo gripper* sedangkan D8 untuk *servo* lengan. Rangkaian servo dengan NodeMCU dapat dilihat seperti gambar 5.5.



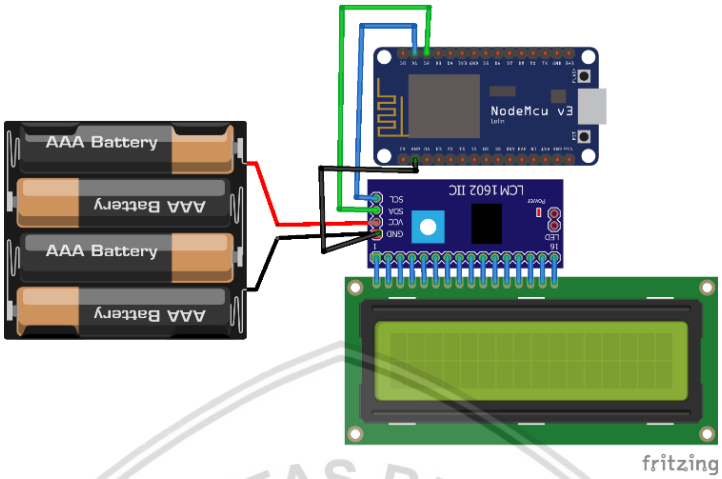
Gambar 5.5 Rangkaian Motor *Servo* dengan *NodeMCU*

Tabel 5.2 Koneksi Pin I2C dengan *NodeMCU*

Motor Servo	*NodeMCU*	Fungsi	Power 5v
SDA	D2	Data	
SCL	D1	Data	
VCC		Power Supply	+
GND	G	Ground	-

Pada tabel 5.2 dapat dilihat bahwa pin VCC pada *I2C* dihubungkan pada tegangan 5v yang dihasilkan dari adaptor. Pin *GND* dihubungkan pada pin G *NodeMCU* dan *ground* adaptor. Pin SDA pada *I2C* dihubungkan ke pin *digital* D2

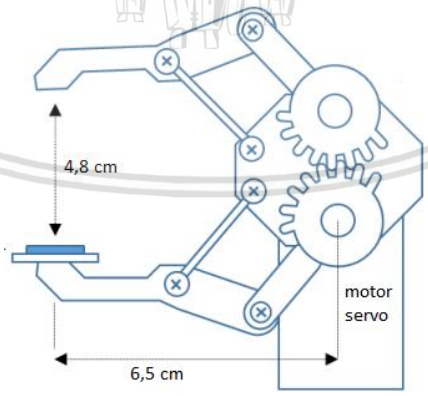
dan Pin SCL pada *I2C* dihubungkan ke pin *digital* D1 pada *NodeMCU*. Lalu, 16 pin yang terdapat pada I2C disambungkan sesuai 16 pin yang terdapat pada LCD seperti pada gambar 5.6.



Gambar 5.6 Rangkaian *LCD I2C* dengan *NodeMCU*

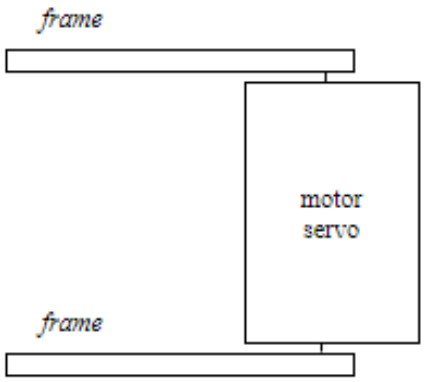
5.1.1.2 Perancangan Servo *Gripper* dan Sensor *FSR*

Gripper yang digunakan adalah model parallel, yakni hanya dengan satu servo bisa langsung menggerakkan dua japit sekaligus. *Gripper* yang bergerak adalah sebelah kiri dengan posisi motorservo berada diatas *gripper*. *Gripper* bisa terbuka sampai 4.8cm sesuai dengan lebar telur ayam pada umumnya sehingga telur ayam dapat mudah masuk. Sensor diletakkan pada sisi japit sebelah kiri, sejajar servo seperti pada gambar 5.7.



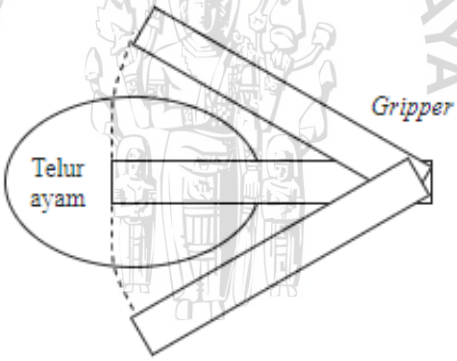
Gambar 5.7 Letak Sensor pada *Gripper*

Selanjutnya, pada bagian belakang *gripper* akan dipasangkan sebuah *frame* yang berfungsi sebagai lengan robot. *Frame* akan bergerak naik turun. Ujung *frame* diletakkan pada tengah samping motor servo seperti gambar 5.8.



Gambar 5.8 Letak *Servo* Belakang sebagai Lengan Robot

Dengan adanya *servo* belakang tersebut, sistem diharapkan dapat dengan tepat menempatkan *end effector gripper* yang telah terpasang sensor sejajar dengan posisi tengah telur. Selain untuk menempatkan ujung *gripper* tepat pada tengah telur ayam, juga dapat dimanfaatkan untuk memindahkan telur ayam baik ke atas maupun kebawah. Gerakan ujung *gripper* jika dilihat dari samping dapat dilihat seperti gambar 5.9.

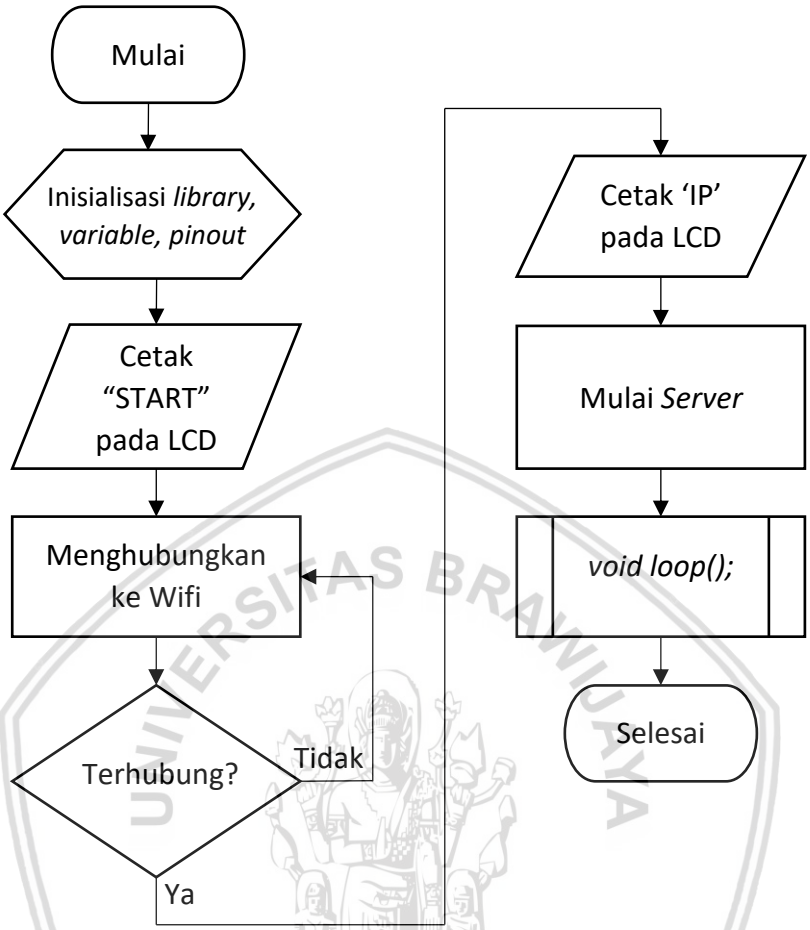


Gambar 5.9 Fungsi Servo Belakang pada *Gripper*

5.1.2 Perancangan Perangkat Lunak

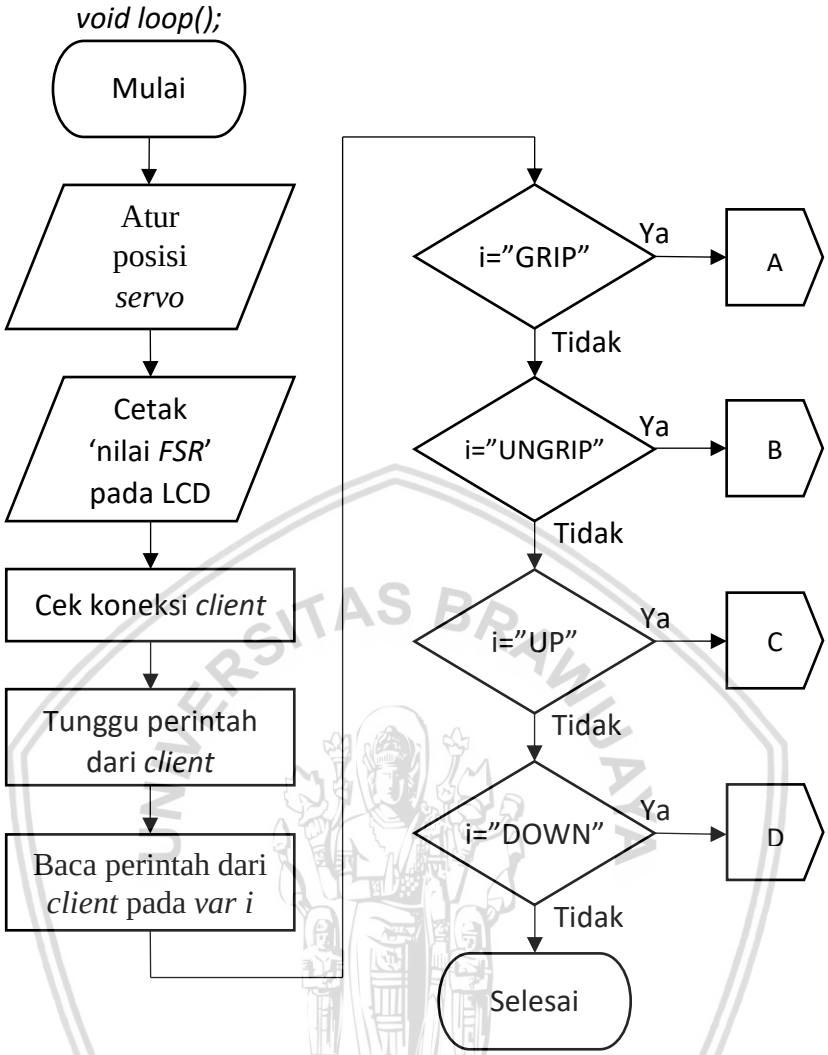
Pada perancangan perangkat lunak terdapat 2 perancangan, yakni perancangan perangkat lunak pada *mikrokontroller* dan perancangan perangkat lunak pada aplikasi Android. Perancangan lebih banyak membahas tentang desain flowchart bagaimana sistem bekerja.

5.1.2.1 Perancangan Perangkat Lunak pada *Mikrokontroller*



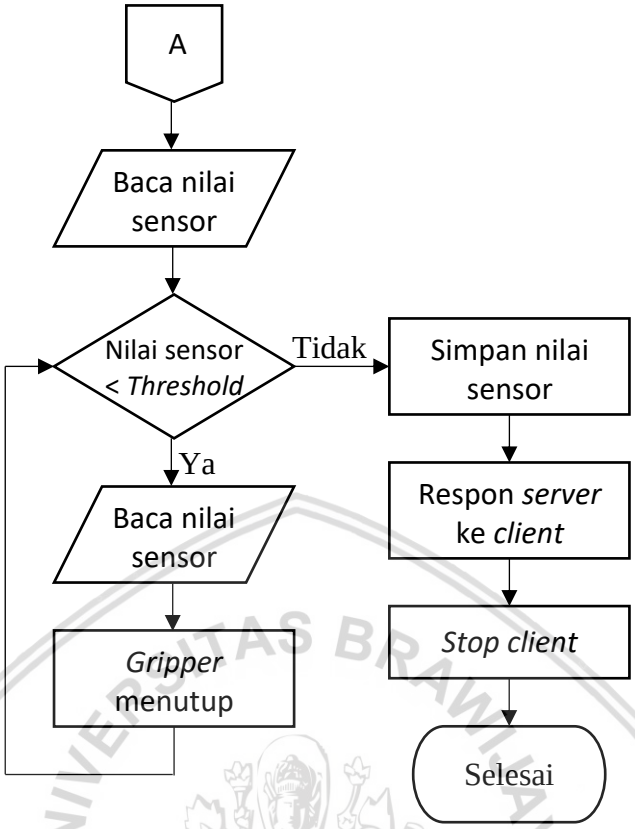
Gambar 5.10 *Flowchart* Perancangan Perangkat Lunak *Mikrokontroller*

Gambar 5.10 menjelaskan tentang alur kerja sistem yang dirancang pada *mikrokontroller* secara keseluruhan. Diawali dari insialisasi *library* yang akan digunakan untuk menjalankan komponen elektronika yang dipakai maupun menyimpan nilai pada *variable*. Lalu ke deklarasi *variable* digunakan untuk menyimpan data *input* dan *output*. Data ini mempunyai tipe data *int* untuk angka, *const int* untuk pin *mikrokontroller*, dan *string* untuk data karakter. Selanjutnya sistem mencetak karakter pada LCD menampilkan kata START sebagai tanda bahwa sistem mulai. Kemudian, sistem menghubungkan ke *wifi* dengan memasukkan *network ssid* dan *password ssid* sebagai autentikasi *client.* Setelah sistem terhubung dengan *wifi,* sistem mendapatkan *IP Address NodeMCU*. *IP Address* ini digunakan sebagai identitas sistem agar bisa terhubung dengan client. Setelah terhubung, sistem menjalankan fungsi loop.



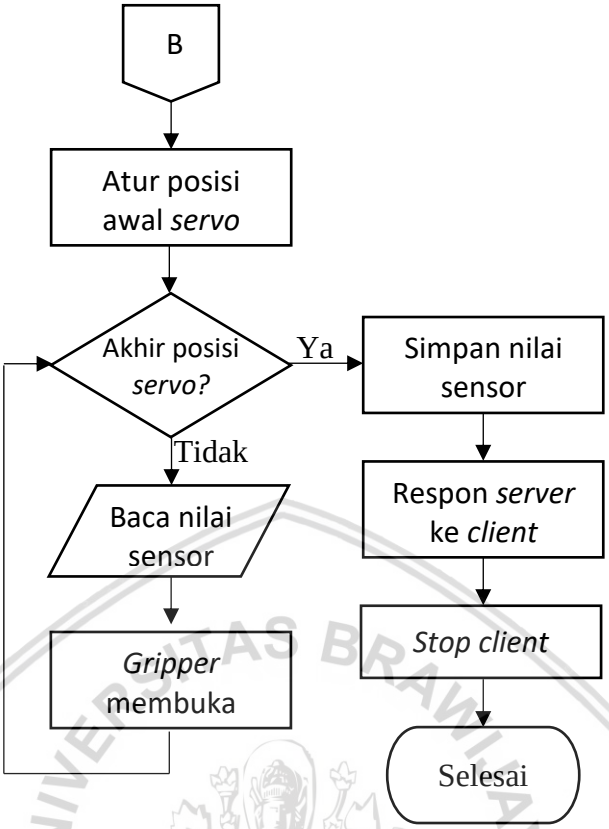
Gambar 5.11 *Flowchart* Perancangan *void loop();*

Fungsi *loop* seperti gambar 5.11 menunjukkan program akan berjalan secara *looping*. Diawali dari mengatur posisi kedua *servo* pada posisi awal program. *Servo gripper* diatur pada kondisi terbuka sedangkan posisi *servo* lengan robot diatur pada posisi lurus kedepan. Selanjutnya, sistem menunggu koneksi dari *client* dan menunggu *client* mengirim *request. Server* siap untuk menerima *request* dari *client*. Setelah *client* mengirim data perintah, *server* membaca perintah tersebut yang disimpan pada *variable i. Variable i* menyimpan karakter *string* sebagai acuan untuk diolah selanjutnya. Setelah *server* menerima perintah dari client, sistem mengolah *variable i* untuk diolah selanjutnya. Adapun 4 kondisi yang dipakai sesuai dengan isi dari *variable i,*yakni GRIP, UNGRIP, UP, dan DOWN. Masing-masing kondisi akan dibahas selanjutnya.



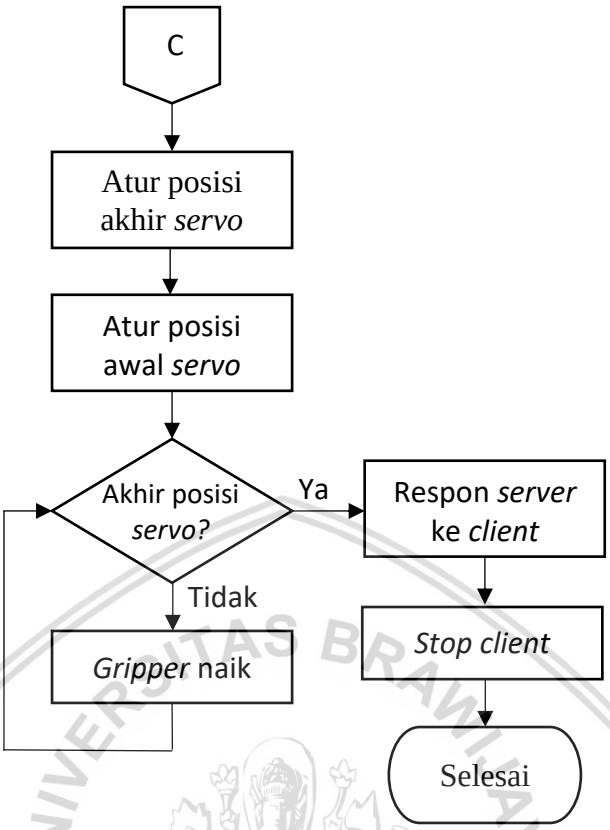
Gambar 5.12 *Flowchart* Fungsi "*GRIP*"

Gambar 5.12 menjelaskan tentang bagaimana fungsi *GRIP* bekerja. Secara sederhana, fungsi *GRIP* bekerja dengan cara membaca nilai sensor dan menggerakkan *servo gripper* secara beriringan. Sebelum *servo* bergerak, ada pembacaan nilai sensor untuk memastikan apakah sensor menekan suatu objek atau tidak. *Servo* bergerak secara menutup dengan bantuan *increment* suatu *variable* sampai posisi *gripper* sampai batas. Pergerakan *servo gripper* dibatasi agar ketika bergerak menutup, sensor tidak membentur ujung *gripper* lainnya. Pembacaan nilai sensor ketika *servo gripper* bergerak dilakukan agar ketika *gripper* bergerak, nilai sensor selalu di perbarui. Pembaruan ini dilakukan agar *servo gripper* dapat dengan tepat berhenti pada waktunya. Setelah nilai sensor melebihi *threshold, servo gripper* akan berhenti bergerak lalu menyimpan nilai sensor terakhir pada suatu *variable.* Nilai *threshold* didapat dari percobaan manual yang akan dibahas pada sub bab 5.1.3. Setelah proses kontrol mulai dari membaca sensor sampai menyimpan nilai sensor terakhir pada suatu *variable* dikerjakan, *server* akan mengirim *response* ke *client* sebagai tanda bukti bahwa tugas yang diberikan sudah dikerjakan. Lalu, *server* memutus koneksi dengan *client.*



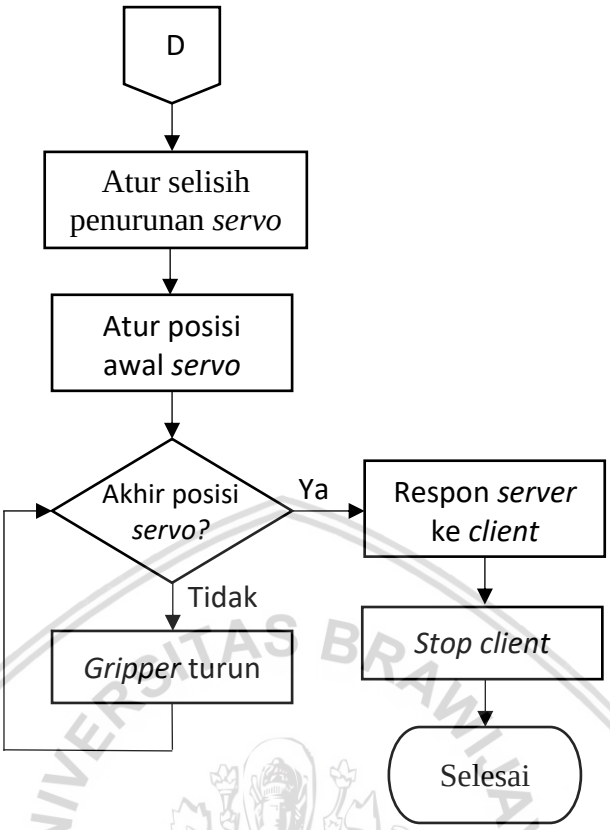
Gambar 5.13 *Flowchart* Fungsi *"UNGRIP"*

Gambar 5.13 menjelaskan tentang bagaimana fungsi UN*GRIP* bekerja. Secara sederhana, fungsi UN*GRIP* bekerja dengan cara menggerakkan *servo gripper* hingga *gripper* terbuka dan kembali ke posisi awal. Gerakan *servo* dibantu dengan suatu *variable* pada program untuk mendefinisikan kemana arah *servo* itu digerakkan. Mulanya, inisialisasi posisi awal *servo gripper.* Inisialisasi *servo* awal dilakukan karena posisi *servo* bisa jadi pada kondisi menutup maupun terbuka. *Servo gripper* akan bergerak jika posisi awal *servo* tidak sama dengan posisi akhir, yakni dalam kondisi posisi *gripper* menutup. Posisi akhir *servo gripper* diatur dalam kondisi terbuka. Posisi *servo* terbuka sudah didefinisikan dari awal pada program nantinya. Lalu, pembacaan nilai sensor ketika *servo gripper* bergerak dilakukan agar ketika *gripper* dalam kondisi terbuka nilai sensor juga kembali ke posisi awal. Setelah proses kontrol mulai dari membaca sensor sampai menyimpan nilai sensor terakhir pada suatu *variable* dikerjakan, *server* akan mengirim *response* ke *client* sebagai tanda bukti bahwa tugas yang diberikan sudah dikerjakan. Lalu, *server* memutus koneksi dengan *client.*



Gambar 5.14 *Flowchart* Fungsi *"UP"*

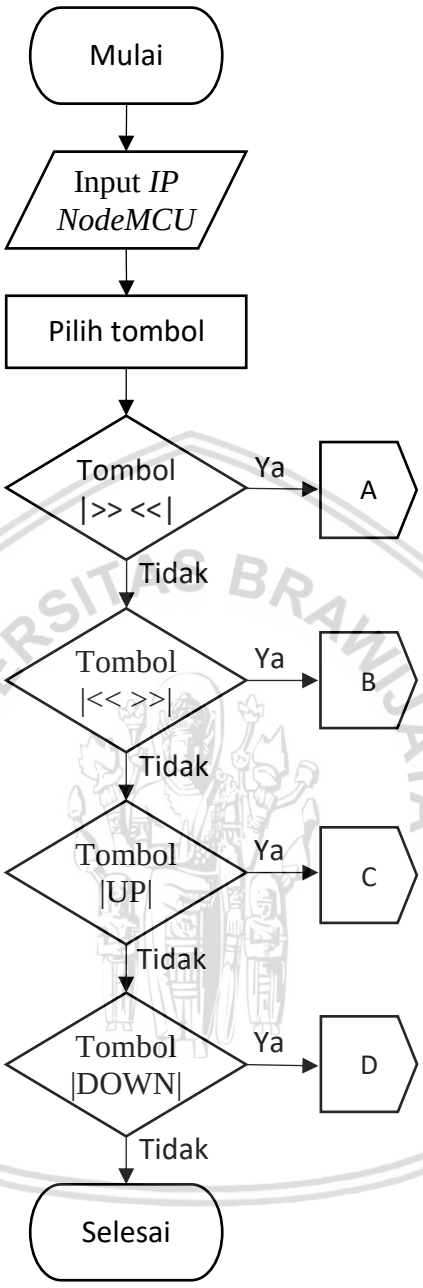
Gambar 5.14 menjelaskan tentang bagaimana fungsi U*P* bekerja. Secara sederhana, fungsi U*P* bekerja dengan cara menggerakkan *servo lengan robot* hingga naik. Gerakan *servo* dibantu dengan suatu *variable* pada program untuk mendefinisikan kemana arah *servo* itu digerakkan. Mulanya, atur posisi akhir *servo lengan robot* agar pergerakan *servo* terarah dan sampai tujuan. Lalu, inisialisasi posisi awal *servo lengan robot. Servo lengan robot* akan bergerak jika posisi awal *servo* tidak sama dengan posisi akhir. Secara perlahan, *servo lengan robot* akan bergerak sampai posisi akhir dengan bantuan *variable servo* yang selalu diperbarui. Pembacaan nilai sensor tidak perlu dilakukan seperti pada kondisi A dan B karena pada kondisi ini lebih menitikberatkan pada posisi *servo lengan robot* yang tidak ada kaitannya dengan sensor. Setelah proses kontrol mulai dari mengatur posisi akhir *servo* sampai *servo* lengan robot naik sesuai tujuan, *server* akan mengirim *response* ke *client* sebagai tanda bukti bahwa tugas yang diberikan sudah dikerjakan. Lalu, *server* memutus koneksi dengan *client.*



Gambar 5.15 *Flowchart* Fungsi "*DOWN*"

Gambar 5.15 menjelaskan tentang bagaimana fungsi DOWN bekerja. Secara sederhana, fungsi DOWN bekerja dengan cara menggerakkan *servo lengan robot* hingga turun. Gerakan *servo* dibantu dengan suatu *variable* pada program untuk mendefinisikan kemana arah *servo* itu digerakkan. Mulanya, atur posisi akhir *servo lengan robot* agar pergerakan *servo* terarah dan sampai tujuan. Lalu, inisialisasi posisi awal *servo lengan robot. Servo lengan robot* akan bergerak jika posisi awal *servo* tidak sama dengan posisi akhir. Secara perlahan, *servo lengan robot* akan bergerak sampai posisi akhir dengan bantuan *variable servo* yang selalu diperbarui. Pembacaan nilai sensor tidak perlu dilakukan seperti pada kondisi A dan B karena pada kondisi ini lebih menitikberatkan pada posisi *servo lengan robot* yang tidak ada kaitannya dengan sensor. Setelah proses kontrol mulai dari mengatur posisi akhir *servo* sampai *servo* lengan robot turun sesuai tujuan, *server* akan mengirim *response* ke *client* sebagai tanda bukti bahwa tugas yang diberikan sudah dikerjakan. Lalu, *server* memutus koneksi dengan *client.*

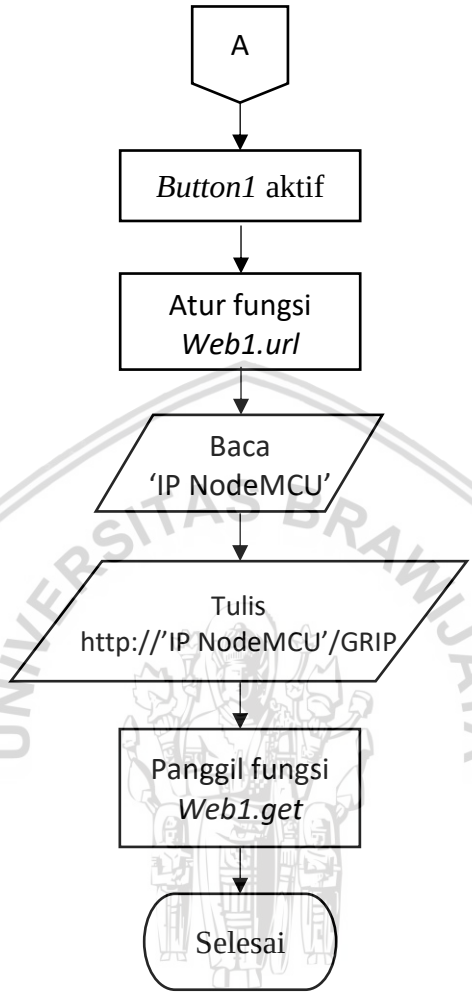
5.1.2.2 Perancangan Perangkat Lunak pada Aplikasi Android



Gambar 5.16 *Flowchart* Aplikasi Android

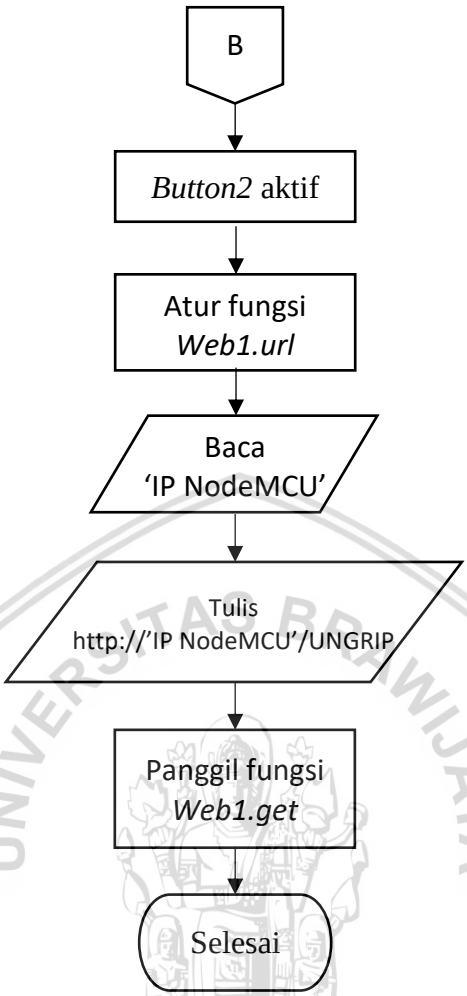
Gambar 5.16 menjelaskan tentang bagaimana aplikasi Android berjalan agar dapat mengontrol *robot*. Awalnya, user memasukkan *IP NodeMCU* yang terbaca pada *LCD I2C NodeMCU* pada aplikasi Android. *IP NodeMCU* digunakan untuk tersambung ke *mikrokontroller NodeMCU* yang berperan sebagai *server.* Selanjutnya, aplikasi Android yang berperan sebagai *client* memilih tombol yang terdapat pada *interface* aplikasi Android. Adapun 4 tombol yang tersedia, yakni tombol |>> <<|, tombol |<< >>|, tombol |UP|, dan tombol |DOWN| dengan

fungsi masing-masing. Semua fungsi masing-masing tombol akan diatur pada *blocks MIT App Inventor*.



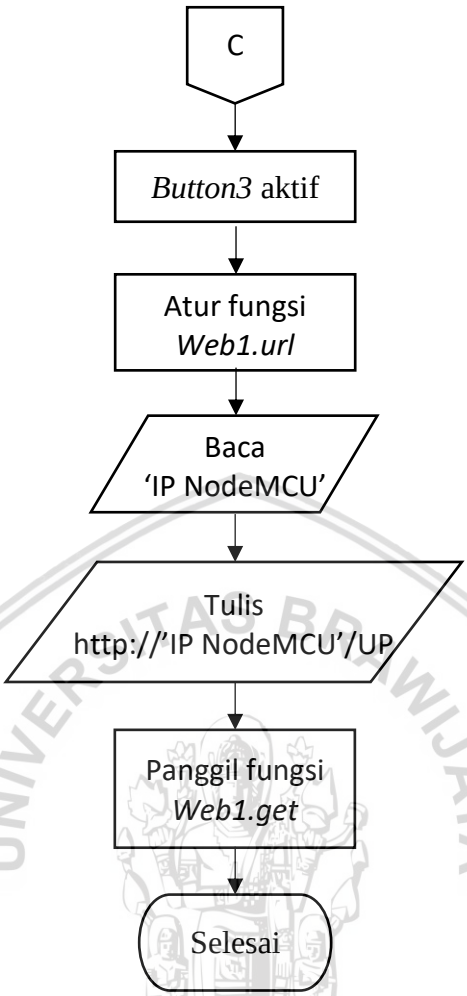
Gambar 5.17 Perancangan Fungsi Button1

Fungsi Button1 pada gambar 5.17 merupakan fungsi yang dijalankan setelah user menekan tombol |>> <<|. Ketika memilih tombol |>> <<|, *variable Button1* aktif. Ketika *variable Button1,* aplikasi Android mengatur fungsi *Web1.url* pada program. Fungsi ini digunakan untuk menuliskan *url* pada *web browser* atau mengirim *request* ke *server NodeMCU*. Pengiriman *request* memerlukan identitas *server.* Identitas *server* yang dimaksud adalah alamat *IP NodeMCU.* Aplikasi Android mendapatkan *IP NodeMCU* dari *input user*. Setelah program membaca *IP NodeMCU, client* mengirim *request url* ke *server.* Misal, *IP NodeMCU* yang dimasukkan adalah 192.168.1.5, maka *request* yang dikirim adalah http://192.168.1.5/GRIP. Setelah itu, *client* mendapatkan *response* dari *server* dengan fungsi *Web1.get* sebagai tanda bukti bahwa request sudah diterima.



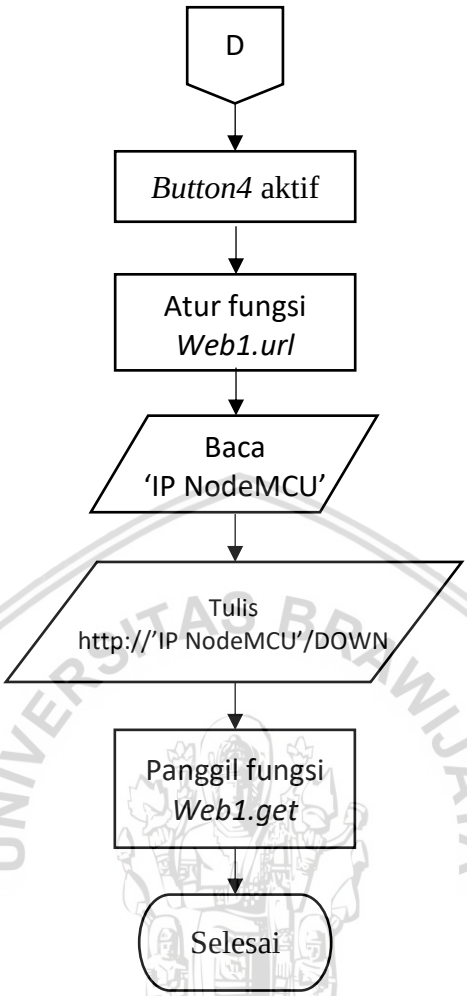
Gambar 5.18 Perancangan Fungsi Button2

Fungsi Button2 pada gambar 5.18 merupakan fungsi yang dijalankan setelah user menekan tombol |<< >>|. Ketika memilih tombol |<< >>|, *variable Button2* aktif. Ketika *variable Button2,* aplikasi Android mengatur fungsi *Web1.url* pada program. Fungsi ini digunakan untuk menuliskan *url* pada *web browser* atau mengirim *request* ke *server NodeMCU*. Pengiriman *request* memerlukan identitas *server.* Identitas *server* yang dimaksud adalah alamat *IP NodeMCU.* Aplikasi Android mendapatkan *IP NodeMCU* dari *input user*. Setelah program membaca *IP NodeMCU, client* mengirim *request url* ke *server.* Misal, *IP NodeMCU* yang dimasukkan adalah 192.168.1.6, maka *request* yang dikirim adalah http://192.168.1.6/UNGRIP. Setelah itu, *client* mendapatkan *response* dari *server* dengan fungsi *Web1.get* sebagai tanda bukti bahwa request sudah diterima.



Gambar 5.19 Perancangan Fungsi Button3

Fungsi Button3 pada gambar 5.19 merupakan fungsi yang dijalankan setelah user menekan tombol |UP|. Ketika memilih tombol |UP|, *variable Button3* aktif. Ketika *variable Button3* aplikasi Android mengatur fungsi *Web1.url* pada program. Fungsi ini digunakan untuk menuliskan *url* pada *web browser* atau mengirim *request* ke *server NodeMCU*. Pengiriman *request* memerlukan identitas *server.* Identitas *server* yang dimaksud adalah alamat *IP NodeMCU.* Aplikasi Android mendapatkan *IP NodeMCU* dari *input user*. Setelah program membaca *IP NodeMCU, client* mengirim *request url* ke *server.* Misal, *IP NodeMCU* yang dimasukkan adalah 192.168.1.7, maka *request* yang dikirim adalah http://192.168.1.7/UP. Setelah itu, *client* mendapatkan *response* dari *server* dengan fungsi *Web1.get* sebagai tanda bukti bahwa request sudah diterima.



Gambar 5.20 Perancangan Fungsi Button4

Fungsi Button4 pada gambar 5.20 merupakan fungsi yang dijalankan setelah user menekan tombol |DOWN|. Ketika memilih tombol |DOWN|, *variable Button4* aktif. Ketika *variable Button4,* aplikasi Android mengatur fungsi *Web1.url* pada program. Fungsi ini digunakan untuk menuliskan *url* pada *web browser* atau mengirim *request* ke *server NodeMCU*. Pengiriman *request* memerlukan identitas *server.* Identitas *server* yang dimaksud adalah alamat *IP NodeMCU.* Aplikasi Android mendapatkan *IP NodeMCU* dari *input user*. Setelah program membaca *IP NodeMCU, client* mengirim *request url* ke *server.* Misal, *IP NodeMCU* yang dimasukkan adalah 192.168.1.8, maka *request* yang dikirim adalah http://192.168.1.8/DOWN. Setelah itu, *client* mendapatkan *response* dari *server* dengan fungsi *Web1.get* sebagai tanda bukti bahwa request sudah diterima.

5.1.2.3 Perancangan *Interface* Aplikasi Android

Aplikasi Android nantinya akan akan digunakan sebagai pengontrol robot. Oleh karena itu, diperlukan desain antarmuka aplikasi agar user dapat menggunakannya lebih mudah. Ada beberapa poin yang akan ditampilkan, yakni judul aplikasi, label untuk *input IP NodeMCU,* dan tombol untuk mengontrol robot. Label paling atas diberi judul GRIPPER CONTROL yang menandakan bahwa user sudah masuk aplikasi. Selanjutnya dibawah label judul adalah kolom untuk *input IP NodeMCU*. Selanjutnya dibawah kolom IP NodeMCU adalah 4 tombol dengan fungsi masing-masing. Adapun 4 tombol itu antara lain tombol |>> <<|, tombol |<< >>|, tombol |UP|, dan tombol |DOWN| yang terletak berdekatan satu sama lain. Nantinya masing –masing tombol ini akan memiliki fungsi yang berbeda. Tombol |>> <<| digunakan untuk mengeratkan *gripper*. Tombol |<< >>| digunakan untuk melonggarkan *gripper*. Tombol |UP| untuk menaikkan lengan robot. Dan tombol |DOWN| untuk menurunkan lengan robot.

5.1.3 Perancangan *Threshold*

Pada perancangan *threshold* merupakan penentuan nilai *threshold* yang akan dipakai pada robot. Nilai *threshold* didapat dari percobaan manual robot. Robot hanya menggunakan sensor dan gripper untuk mengetahui seberapa erat kekuatan sensor pada satu buah telur ayam. Percobaan dilakukan sebanyak 11 kali dengan ketentuan memegang mulai dari keeratan yang longgar sampai keeratan yang sangat erat. Dari beberapa percobaan didapat nilai keeratan seperti pada tabel 5.3. Setelah mendapat 11 data, dicari nilai rerata dan hasil rerata akan dijadikan sebagai nilai *threshold*.

Tabel 5.3 Akuisisi Data Nilai Sensor pada Sebuah Telur Ayam

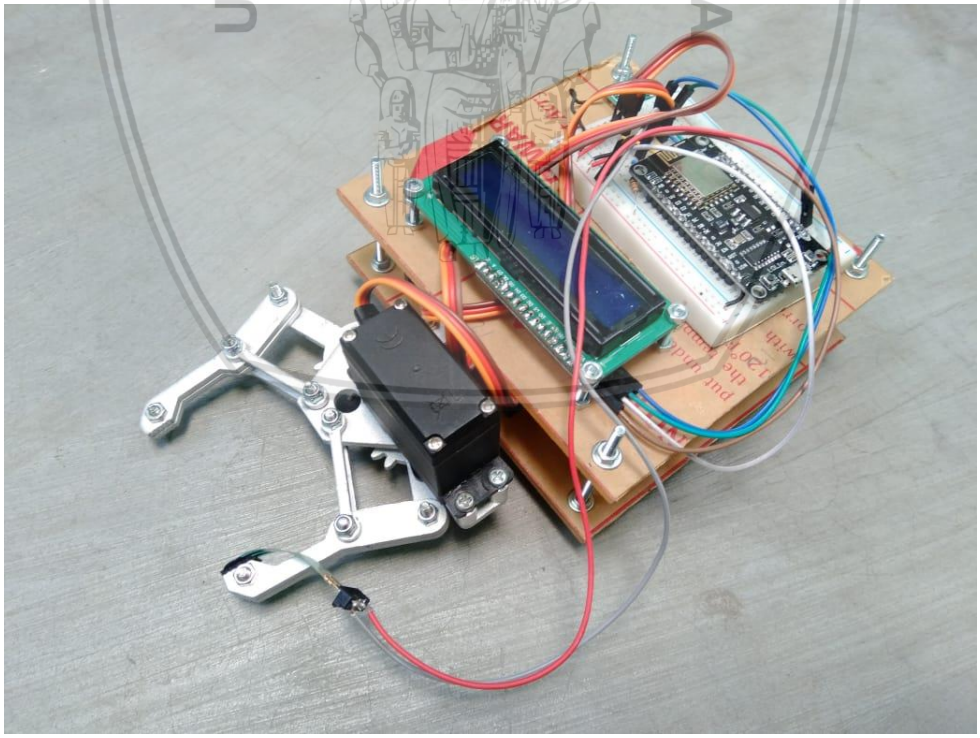
Percobaan Ke-	Nilai Sensor
1	326
2	497
3	594
4	636
5	627
6	578
7	609
8	641
9	626
10	558
11	386
Rerata	552

5.2 Implementasi Sistem

Pada tahap ini akan dijelaskan mengenai implementasi perangkat keras dan perangkat lunak pada sistem dan aplikasi Android. Implementasi didasar atas perancangan yang sudah dibuat.

5.2.1 Implementasi Perangkat Keras

Implementasi perangkat keras seperti gambar 5.21 merupakan wujud dari semua yang telah dilaksanakan sesuai dengan perancangan. Implementasi dikerjakan sesuai dengan perancangan untuk meminimalisir kesalahan implementasi. Sistem memiliki 2 buah servo sebagai penggerak *gripper* dan lengan robot. Implementasi mempunyai 2 fungsi motor *servo*, yakni untuk menggerakkan *end effector gripper* dan naik turun *gripper*. Semua fungsi dilakukan oleh *servo* yang terpasang dan dikontrol melalui perangkat Android. Sensor *FSR (Force Sensitive Resistor)* dipasang pada ujung *gripper* agar sensor dapat berentuhan langsung dengan objek telur ayam. Rangkaian elektronik secara keseluruhan diletakkan pada bagian atas atau lantai atas agar *NodeMCU* dapat dengan mudah menangkap sinyal *wifi* karena berada pada tempat terbuka. Pada bagian bawah atau lantai bawah terdapat motor *servo* yang berfungsi sebagai lengan robot.



Gambar 5.21 Implementasi Perangkat Keras Sistem

5.2.2 Implementasi Perangkat Lunak

Pada implementasi perangkat lunak terdapat 2 bagian, yakni implementasi perangkat lunak pada *mikrokontroller* dan implementasi perangkat lunak Android.

5.2.2.1 Implementasi Perangkat Lunak *Mikrokontroller*

Agar dapat menggunakan *wifi*, *servo*, dan *LCD I2C* maka perlu adanya penggunaan library dan penyesuaian *variable* seperti pada tabel 5.4.

Tabel 5.4 *Source Code* Deklarasi *library*

No	Kode Program
1	`#include <ESP8266WiFi.h>`
2	`#include <Servo.h>`
3	`#include <Wire.h>`
4	`#include <LiquidCrystal_I2C.h>`
5	
6	`WiFiServer server(80);`
7	`Servo servoFront;`
8	`Servo servoBack;`
9	`LiquidCrystal_I2C lcd(0x3F ,2,1,0,4,5,6,7,3, POSITIVE);`

Deklarasi library pada *NodeMCU* dilakukan agar sistem dapat mengenali komponen yang telah tersambung. Seperti *library* ESP8266WiFi.h digunakan untuk koneksi *NodeMCU* ke *wifi* yang tersedia. *Library* Servo.h digunakan untuk mengendalikan motor *servo*. *Library* Wire.h digunakan untuk komunikasi *serial* pada *adapter* I2C. *Library* LiquidCrystal_I2C.h digunakan untuk menampilkan karakter pada LCD yang telah tersambung dengan *adapter* I2C.

Instance dari *library* WiFiServer adalah server(80). Artinya, *variable* bernama server dan menggunakan *port* 80 atau protokol http. *Instance* servoFront dan servoBack digunakan untuk menjalankan fungsi-fungsi yang terdapat pada *library* servo.h. *Instance* servoFront digunakan untuk servo *gripper* sedangkan servoBack digunakan untuk lengan *gripper*. *Instance* lcd(0x3F ,2,1,0,4,5,6,7,3, POSITIVE) digunakan untuk menjalankan fungsi *LCD* yang sudah terhubung dengan *adapter* I2C. Alamat yang dipakai pada *LCD I2C* adalah 0x3F. Alamat ini didapat dari adapter tipe PCF8574AT dengan koneksi A0, A1, dan A2 adalah 0. Untuk angka 2,1,0,4,5,6,7,3 menunjukkan pin yang dipakai pada LCD, yakni pin En, Rw, Rs, d4, d5, d6, d7, dan backlightpin. Kata POSITIVE digunakan untuk *polaritas* LCD.

Tabel 5.5 *Source Code* Deklarasi *Variable Global* dan *pinOut*

No	Kode Program
11	`String  i;`
12	`int fsr = 0;`

```
13  int fsrEnd = 0;
14  int posFront = 45;
15  int posBack = 45;
16  const int pinServoFront = D4;
17  const int pinServoBack = D8;
```

Tabel 5.5 merupakan kode program yang menunjukkan *variable global*. *Variable global* digunakan agar mudah diakses oleh fungsi-fungsi program dibawahnya. *Variable* i bertipe *string* digunakan untuk akses karakter pada *url web*. *Variable* fsr bertipe *integer* digunakan untuk mendapatkan nilai sensor dimulai dari 0. *Variable* fsrEnd bertipe *integer* digunakan untuk menyimpan nilai sensor yang terakhir didapatkan ketika sensor telah mencengkram objek. *Variable* posFront dan posBack bertipe *integer* digunakan untuk menempatkan posisi awal servo yakni pada 45^0. *Variable* pinServoFront digunakan untuk deklarasi pin yang dipakai oleh data servo depan, yakni pada pin D4. Sedangkan pada servo belakang, variable pinServoBack menggunakan pin D8.

Tabel 5.6 *Source Code* Fungsi void setup();

No	Kode Program

```
19  void setup() {
20    i = "";
21    lcd.begin (16,2);
22    lcd.clear();
23    lcd.setCursor(5, 0);
24    lcd.print("START");
25    lcd.setCursor(2, 1);
26    lcd.print("GRIPPER ROBOT");
27    delay(3000);
28    servoFront.attach(pinServoFront);
29    servoBack.attach(pinServoBack);
30    WiFi.disconnect();
31    delay(3000);
32    WiFi.begin("ZB555KL","rahasiaa");
33    while ((!(WiFi.status() == WL_CONNECTED))){
34      delay(300);
35    }
36    lcd.clear();
37    lcd.setCursor(0, 0);
38    lcd.print("Connected");
39    lcd.setCursor(0, 1);
40    lcd.print((WiFi.localIP().toString()));
41    server.begin();
42    delay(3000);
43  }
```

Pada tabel 5.6 merupakan kode program yang menunjukkan bagaimana fungsi *void setup();* berjalan. Dimulai deklarasi *variable* i dengan tanda "" yang menunjukkan bahwa dia bertipe *string*. Kemudian deklarasi lcd.begin(16x2) untuk memulai menjalankan lcd dengan ukuran 16x2. Kemudian, deklarasi lcd.clear() untuk menghapus semua karakter yang pernah ditampilkan sebelumnya, agar karakter tidak saling menimpa. Kemudian, lcd.setCursor() digunakan untuk meletakkan awal karakter ditampilkan, sedangkan lcd.print() digunakan untuk menulis karakter apa yang ingin ditampilkan. Seperti kode program diatas yang menunjukkan sistem awalnya menampilkan kata START pada kolom 5 baris 0 dan kata GRIPPER ROBOT pada kolom 2 baris 1. Karakter ini ditampilkan selama 3 detik agar lebih mudah ditandai bahwa sistem sudah mulai berfungsi.

Fungsi *attach* pada kedua *servo* ini menunjukkan pin yang akan digunakan pada *variable* tersebut. Seperti kode program diatas. Servo depan pada *variable* servoFront terletak pada pin D4 NodeMCU. Servo belakang pada *variable* servoBack terletak pada pin D8.

Kemudian dilanjut koneksi sistem dengan *wifi*. Mulai fungsi *disconnect* agar *wifi* tidak tersambung dengan *wifi* sebelumnya. Koneksi *wifi* menggunakan fungsi Wifi.begin() dengan memasukkan *ssid* dan *password*. *Ssid* yang digunakan pada program ini adalah ZB555KL dan *password* adalah rahasiaa. Pada baris 37-40 merupakan proses pengkoneksian *NodeMCU* dengan *wifi*. Pengecekan dilakukan secara berulang dengan membandingkan apakah status *wifi* sudah terkoneksi atau belum. Proses perulangan koneksi ini dilakukan dengan *delay* 300ms.

Setelah melewati baris 40, artinya NodeMCU sudah terkoneksi *wifi*. Kemudian sistem menandai dengan menampilkan kata Connected pada LCD kolom 0 baris 0. Lalu menampilkan *IP NodeMCU* pada kolom 0 baris 1 dengan memanfaatkan fungsi WiFi.localIP(). Nilai WiFi.localIP() diubah menjadi *string* agar lebih mudah ditampilkan pada LCD. Setelah menampilkan *IP NodeMCU* server siap dimulai dengan memanfaatkan fungsi server.begin(). *IP NodeMCU* ditampilkan pada LCD selama 3 detik agar *user* dapat dengan mudah mencatat *IP NodeMCU* pada aplikasi Android.

Tabel 5.7 *Source Code* Fungsi *void loop();*

No	Kode Program
45	`void loop() {`
46	`  servoFront.write(posFront);`
47	`  delay(15);`
48	`  servoBack.write(posBack);`
49	`  delay(15);`
50	`  lcd.clear();`
51	`  lcd.setCursor(0, 0);`

```
52    lcd.print("Pressure:");
53    lcd.setCursor(9,0);
54    lcd.print(fsrEnd);
55    lcd.setCursor(0, 1);
56    lcd.print("Servo(f,b):");
57    lcd.setCursor(11,1);
58    lcd.print(posFront);
59    lcd.setCursor(13,1);
60    lcd.print(",");
61    lcd.setCursor(14,1);
62    lcd.print(posBack);
63    WiFiClient client = server.available();
64    if (!client) { return; }
65    while(!client.available()){  delay(1); }
66    i = (client.readStringUntil('\r'));
67    i.remove(0, 5);
68    i.remove(i.length()-9,9);
--  ----------------------------------------
144 }
```

Tabel 5.7 merupakan kode program yang menunjukkan bagaimana fungsi void loop(); berjalan. Mulanya, program mengatur posisi *servo* sesuai dengan nilai derajat yang telah ditentukan di awal. Nilai derajat posisi *servo* disimpan pada *variable* posBack dan posFront. Kemudian, menampilkan informasi pada LCD berupa nilai terakhir sensor yang disimpan pada *variable* fsrEnd dan *variable posBack posFront* sebagai petunjuk posisi *servo* depan dan belakang. Perlu diketahui beda *variable* fsr dan fsrEnd adalah fsr untuk mendapatkan nilai sensor sedangkan fsrEnd untuk menyimpan nilai sensor yang terakhir didapatkan.

Kemudian, *server* NodeMCU menunggu koneksi dari *client* untuk bersiap menerima *request.* Proses menunggu koneksi dari *client* ditunjukkan pada baris 63. Proses menunggu client mengirim data ditunjukkan pada baris 64-65. Data dari *client* yang dikirim akan disimpan pada *variable* i. Setelah ada data yang dikirim oleh *client, server* membaca data tersebut. *Variable* i didapat dari *url* pada *web* setelah *IP Address*. Misal, *url web* adalah http://192.168.43.46/GRIP maka *variable i* berisi GRIP. Mulanya, *variable* i yang lama dihapus terlebih dahulu agar tidak tertimpa dengan *variable* i yang baru. Setelah *variable* i sudah terbaca, dilanjutkan ke fungsi kondisional *variable* i. Mulai dari baris 69 sampai 143 merupakan kondisional *variable* i.

Tabel 5.8 Kondisi *Variable* i = "GRIP"

No	Kode Program
69	`  if(i == "GRIP") {`
70	`    fsr = analogRead(A0);`
71	`    while(fsr < 552) {`
72	`      fsr = analogRead(A0);`
73	`      if (posFront >= 75) {`
74	`        posFront--;`
75	`      } else {`
76	`        posFront++;`
77	`        servoFront.write(posFront);`
78	`        delay(15);`
79	`      }`
80	`    }`
81	`    fsrEnd = fsr;`
82	`    client.println("HTTP/1.1 200 OK");`
83	`    client.println("Content-Type: text/html");`
84	`    client.println("");`
85	`    client.println("<!DOCTYPE HTML>");`
86	`    client.println("<html>");`
87	`    client.println("ERAT");`
88	`    client.println("</html>");`
89	`    client.stop();`
90	`    delay(1);`
91	`  }`

Tabel 5.8 merupakan kode program yang menunjukkan bagaimana ketika *variable* i = GRIP. Mulanya pada kondisi ini *variable* fsr membaca nilai sensor untuk mengetahui berapakah nilai sensor pada saat itu. Nilai sensor nantinya akan dijadikan sebagai acuan untuk dibandingkan dengan *threshold.*

Threshold yang dipakai pada program adalah 552. Angka 552 didapat dari hasil rerata pada 11 akuisisi data nilai sensor menggunakan sensor FSR. Dari 11 percobaan akuisisi data sensor didapat hasil nilai sensor antara lain 326, 497, 594, 636, 627, 578, 609, 641, 626, 558, 386. Dari 11 angka tersebut didapat hasil rata-rata adalah 552. Kemudian angka 552 dijadikan *threshold*. Perlu diketahui bahwa, dalam mencari data *threshold* ini memakai kode program yang berbeda. Yakni dengan memanfaatkan fungsi analogRead() dengan *delay* 1 detik.

Ketika nilai fsr belum mencapai *threshold* 552 maka kode program masuk pada fungsi perulangan untuk menggerakkan *servo gripper*. *Gripper* akan terus menutup dengan fungsi *increment* pada *variable pos*. Ketika *gripper* sedang bergerak menutup, sistem juga membaca nilai sensor yang disimpan sementara pada *variable fsr*. Hal ini dilakukan agar nilai sensor dapat dipantau secara realtime ketika *gripper* bergerak menutup. Sensor akan semakin bertambah besar karena

semakin *gripper* menutup maka semakin rekat pula *gripper* menjapit objek telur ayam. Ketika *variable fsr* sudah melewati *threshold* 552, maka sistem akan keluar dari fungsi perulangan dan menyimpan nilai fsr terakhir pada *variable* fsrEnd. Setelah melaksanakan tugas, *server NodeMCU* memberikan *response* ke *client* dengan menjawab OK dan menampilkan kata ERAT pada *web*.

Tabel 5.9 Kondisi *Variable* i = "UNGRIP"

No	Kode Program
92	`if(i == "UNGRIP") {`
93	`for(int pos = posFront ; pos >= 45 ; pos--) {`
94	`fsr = analogRead(A0);`
95	`servoFront.write(pos);`
96	`delay(15);`
97	`posFront = pos;`
98	`}`
99	`fsrEnd = fsr;`
100	`client.println("HTTP/1.1 200 OK");`
101	`client.println("Content-Type: text/html");`
102	`client.println("");`
103	`client.println("<!DOCTYPE HTML>");`
104	`client.println("<html>");`
105	`client.println("LONGGAR");`
106	`client.println("</html>");`
107	`client.stop();`
108	`delay(1);`
109	`}`

Tabel 5.9 merupakan kode program yang menunjukkan bagaimana ketika *variable* i = UNGRIP. Secara keseluruhan fungsi ini mengembalikan posisi *servo gripper* kembali ke posisi awal program dengan tetap membaca nilai sensor sebagai tanda bahwa nilai sensor berkurang dan sudah melepaskan objek. Ketika mengerjakan kondisi ini, bisa jadi kondisi *gripper* sedang dalam mencengkram objek. Jadi, kode program langsung pada fungsi perulangan gerakan *servo*. Gerakan *servo* pada *variable* pos dimulai dari *variable* posFront atau posisi akhir *servo gripper*. *Servo* bergerak menuju posisi awal program yakni 45^0. Gerakan posisi *servo* dilakukan dengan bantuan *decrement variable* pos. Selama perulangan *servo* kembali ke posisi awal, *variable* posFront akan selalu di update dari hasil *variable* pos guna pada perulangan motor *servo* tidak kembali ke awal definsi perulangan. Setelah *servo* kembali ke posisi awal, nilai sensor yang terdapat pada *variable* fsr disimpan pada *variable* fsrEnd guna sebagai tanda bahwa sensor sudah melepaskan objek. Setelah melaksanakan tugas, *server NodeMCU* memberikan *response* ke *client* dengan menjawab OK dan menampilkan kata LONGGAR pada *web*.

Tabel 5.10 Kondisi *Variable* i = "UP"

No	Kode Program
110	`  if (i == "UP") {`
111	`    int posBackDown = posBack - 3;`
112	`    for(int pos = posBack ; pos >= posBackDown ; pos-`
113	`-) {`
114	`      servoBack.write(pos);`
115	`      delay(15);`
116	`      posBack = pos;`
117	`    }`
118	`    client.println("HTTP/1.1 200 OK");`
119	`    client.println("Content-Type: text/html");`
120	`    client.println("");`
121	`    client.println("<!DOCTYPE HTML>");`
122	`    client.println("<html>");`
123	`    client.println("NAIK");`
124	`    client.println("</html>");`
125	`    client.stop();`
126	`    delay(1);`
127	`  }`

Pada tabel 5.10 merupakan kode program yang menunjukkan bagaimana ketika *variable* i = UP. Secara keseluruhan, pada kondisi ini posisi lengan *gripper* naik 3 derajat. Proses bermula dari deklarasi selisih derajat kenaikan yakni 3 derajat yang disimpan pada *variable* posBackDown. Lalu langsung masuk ke perulangan gerakan *servo*. Pada gerakan naik *servo* ini, posisi *servo* dimulai dari posisi *variable* posBack yang disimpan pada *variable* pos. Lalu *variable* pos bergerak secara *decrement* sampai posisi *variable* posBackDown yang telah di proses diatas. Selama perulangan, *variable* posBack akan selalu diperbarui sesuai dengan posisi *variable* pos. Sehingga perulangan berjalan secara terurut. Setelah melaksanakan tugas, *server NodeMCU* memberikan *response* ke *client* dengan menjawab OK dan menampilkan kata NAIK pada *web*.

Tabel 5.11 Kondisi *Variable* i = "DOWN"

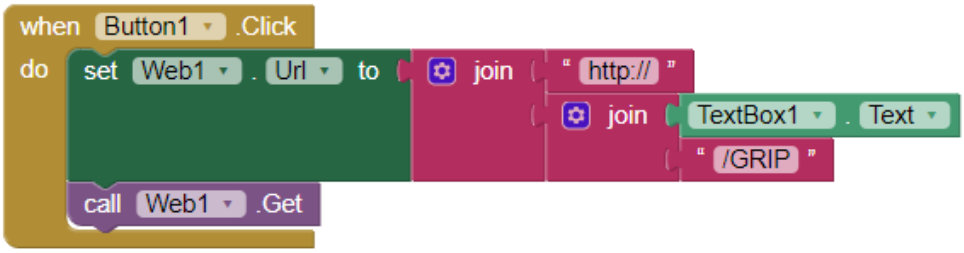
No	Kode Program
127	`  if (i == "DOWN") {`
128	`    int posBackUp = posBack + 3;`
129	`    for(int pos = posBack ; pos <= posBackUp ; pos++)`
130	`{`
131	`      servoBack.write(pos);`
312	`      delay(15);`
133	`      posBack = pos;`
134	`    }`
135	`    client.println("HTTP/1.1 200 OK");`
136	`    client.println("Content-Type: text/html");`

```
137       client.println("");
138       client.println("<!DOCTYPE HTML>");
139       client.println("<html>");
140       client.println("TURUN");
141       client.println("</html>");
142       client.stop();
143       delay(1);
        }
```

Pada tabel 5.11 merupakan kode program yang menunjukkan bagaimana ketika *variable* i = DOWN. Secara keseluruhan, pada kondisi ini posisi lengan *gripper* turun 3 derajat. Proses bermula dari deklarasi selisih derajat penurunan yakni 3 derajat yang disimpan pada *variable* posBackUp. Lalu langsung masuk ke perulangan gerakan *servo*. Pada gerakan naik *servo* ini, posisi *servo* dimulai dari posisi *variable* posBack yang disimpan pada *variable* pos. Lalu *variable* pos bergerak secara *increment* sampai posisi *variable* posBackUp yang telah di proses diatas. Selama perulangan, *variable* posBack akan selalu diperbarui sesuai dengan posisi *variable* pos. Sehingga perulangan berjalan secara terurut. Setelah melaksanakan tugas, *server NodeMCU* memberikan *response* ke *client* dengan menjawab OK dan menampilkan kata TURUN pada *web*.

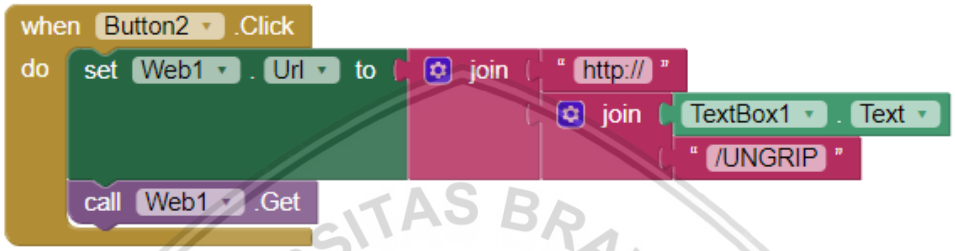
5.2.2.2 Implementasi Perangkat Lunak Android

Pada pembuatan aplikasi Android dengan *MIT App Inventor*, pengguna dimudahkan dengan hanya membuat antarmuka aplikasi, fungsi tiap tombol, label, textbox dan pengaturan fungsi tersebut melalui desain *blocks*. Aplikasi pengontrol sistem akan dinamai AppGripper. Pada aplikasi AppGripper ini mempunyai beberapa button, label dan textbox dengan variabe yang berbeda. Adapun beberapa hal fungsi utama yang akan diimplementasikan untuk membuat aplikasi AppGripper, yakni label, textbox, button, dan web. Label digunakan sebagai tanda informasi. Textbox digunakan untuk kolom *input IP NodeMCU*. Button digunakan untuk *input* pengontrol. Dan web digunakan untuk akses *url* pada suatu web. Semua fungsi tiap tombol dirangkai pada *blocks* yang telah dijelaskan pada perancangan.



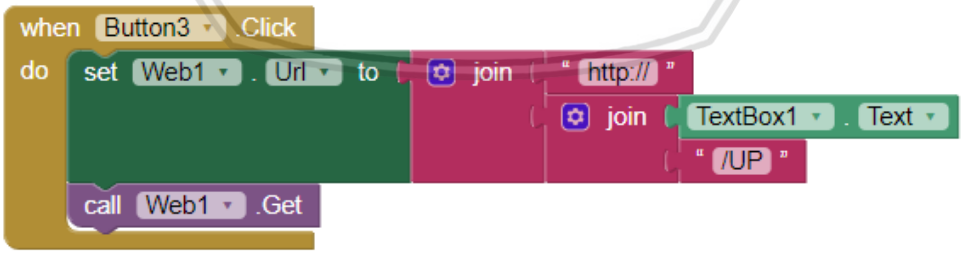
Gambar 5.22 Implementasi *Variable Button1*

Variable Button1 merupakan identitas dari tombol |>> <<|. Seperti gambar 5.22, *variable Button1* digunakan untuk mengetikkan *url* pada suatu *web* secara otomatis. *Url* ini merupakan *request* yang dikirimkan oleh *client* kepada *server*. Dalam pembuatan *url* mulanya dari fungsi join dari beberapa *text*. *Text* yang dijoinkan antara lain *text* "http://" lalu *text* yang ada pada *variable* TextBox1 lalu *text* /GRIP. Misal, url yang terbentuk adalah http://192.168.43.48/GRIP. *Text* ini di set pada Web1 sebagai link url. Setelah *request url* dari *client* dikirimkan ke *server*, *server* akan memberikan respon ke *client* dan *response* akan dibaca dengan menggunakan fungsi Web1.Get.



Gambar 5.23 Implementasi *Variabel Button2*

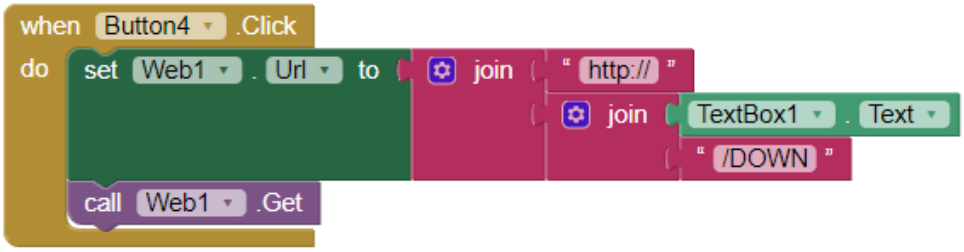
Variable Button2 merupakan identitas dari tombol |<< >>|. Seperti gambar 5.23, *variable Button2* digunakan untuk mengetikkan *url* pada suatu *web* secara otomatis. *Url* ini merupakan *request* yang dikirimkan oleh *client* kepada *server*. Dalam pembuatan *url* mulanya dari fungsi join dari beberapa *text*. *Text* yang dijoinkan antara lain *text* "http://" lalu *text* yang ada pada *variable* TextBox1 lalu *text* /UNGRIP. Misal, url yang terbentuk adalah http://192.168.43.48/UNGRIP. *Text* ini di set pada Web1 sebagai link url. Setelah *request url* dari *client* dikirimkan ke *server*, *server* akan memberikan respon ke *client* dan *response* akan dibaca dengan menggunakan fungsi Web1.Get.



Gambar 5.24 Implementasi *Variabel Button3*

Variable Button3 merupakan identitas dari tombol |>> <<|. Seperti gambar 5.24, *variable Button3* digunakan untuk mengetikkan *url* pada suatu *web* secara otomatis. *Url* ini merupakan *request* yang dikirimkan oleh *client* kepada *server*. Dalam pembuatan *url* mulanya dari fungsi join dari beberapa *text*. *Text* yang dijoinkan antara lain *text* "http://" lalu *text* yang ada pada *variable* TextBox1 lalu *text* /UP. Misal, url yang terbentuk adalah http://192.168.43.48/UP. *Text* ini di set

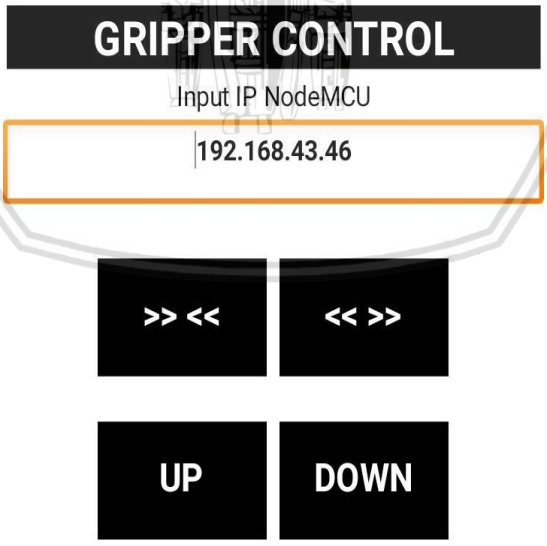
pada Web1 sebagai link url. Setelah *request url* dari *client* dikirimkan ke *server*, *server* akan memberikan respon ke *client* dan *response* akan dibaca dengan menggunakan fungsi Web1.Get.



Gambar 5.25 Implementasi *Variabel Button4*

Variable Button4 merupakan identitas dari tombol |>> <<|. Seperti gambar 5.25, *variable Button4* digunakan untuk mengetikkan *url* pada suatu *web* secara otomatis. *Url* ini merupakan *request* yang dikirimkan oleh *client* kepada *server*. Dalam pembuatan *url* mulanya dari fungsi join dari beberapa *text*. *Text* yang dijoinkan antara lain *text* "http://" lalu *text* yang ada pada *variable* TextBox1 lalu *text* /DOWN. Misal, url yang terbentuk adalah http://192.168.43.48/DOWN. *Text* ini di set pada Web1 sebagai link url. Setelah *request url* dari *client* dikirimkan ke *server*, *server* akan memberikan respon ke *client* dan *response* akan dibaca dengan menggunakan fungsi Web1.Get.

5.2.3 Implementasi *Interface* Aplikasi Android



Gambar 5.26 Implementasi *Interface* Aplikasi Android

Gambar 5.26 merupakan implementasi *interface* aplikasi Android yang telah dibuat sesuai perancangan. Terdapat label GRIPPER CONTROL sebagai penanda bahwa *user* sudah masuk aplikasi. Dibawah label ada *textbox* yang difungsikan untuk *input IP NodeMCU* pada kolom kosong. Misal *IP NodeMCU* diatas adalah 192.168.43.48. Dibawah kolom *IP NodeMCU* terdapat 4 jenis tombol yang berbeda. Semua tombol mempunyai fungsi yang berbeda. Cara menekan tombol cukup sekali tekan, bukan ditekan lama terus menerus. Hanya sekali tekan, sistem sudah melaksanakan perintah. Fungsi tombol |>> <<| digunakan untuk mengeratkan *gripper.* Fungsi tombol |<< >>| digunakan untuk melonggarkan *gripper.* Fungsi tombol |UP| digunakan untuk menaikkan *gripper.* Dan fungsi tombol |DOWN| digunakan untuk menurunkan *gripper.*

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini akan dilakukan pengujian sekaligus analisis dari sistem yang sudah dibuat. Pengujian terdiri dari 3 bagian, yakni pengujian pengiriman perintah, pengujian akuisisi data nilai sensor, pengujian *gripper* berhenti otomatis terhadap *threshold*.

6.1 Pengujian Pengiriman Perintah

6.1.1 Tujuan Pengujian

Pengujian ini dilakukan untuk mengetahui keberhasilan mengirim perintah dari perangkat Android ke *NodeMCU*. Adapun 4 perintah yang digunakan, yakni mengeratkan, melonggarkan, naik dan turun. Mengeratkan dan melonggarkan digunakan untuk memegang objek. Sedangkan, naik turun digunakan untuk memindahkan objek atau untuk mengatur posisi tengah objek. Keberhasilan pengiriman perintah ditandai dari robot menerima perintah tersebut.

6.1.2 Prosedur Pengujian

Adapun prosedur pengujian pengiriman printah untuk mengontrol robot adalah sebagai berikut:

1. Masuk pada aplikasi Android.
2. Memasukkan *IP Address NodeMCU* pada tampilan aplikasi perangkat Android.
3. Menekan tombol sekali secara berurutan dan bergantian mulai dari “”>> <<”, “<< >>”, “UP”, “DOWN” sebanyak 5 kali.

6.1.3 Hasil Pengujian

Tabel 6.1 merupakan hasil pengujian pengiriman perintah kendali *gripper*. Pengiriman perintah dilakukan dari perangkat Android ke sistem mikrokontroller melalui sambungan *wifi*.

Tabel 6.1 Hasil Pengujian Pengiriman Perintah Kendali *Gripper*

No	Perintah Tombol	Pengujian ke-	Respon Sistem	Hasil
1	>> <<	1	*Gripper* menutup	Berhasil
		2	Gripper menutup	Berhasil
		3	Gripper menutup	Berhasil
		4	Gripper menutup	Berhasil
		5	Gripper menutup	Berhasil

2	<< >>	1	*Gripper* membuka	Berhasil
		2	*Gripper* membuka	Berhasil
		3	*Gripper* membuka	Berhasil
		4	*Gripper* membuka	Berhasil
		5	*Gripper* membuka	Berhasil

Pada tabel 6.1 dapat dilihat bahwa ada dua perintah yang digunakan yakni tombol ">> <<" sebagai mengeratkan *gripper* dan "<< >>" sebagai melonggarkan *gripper*. Dalam sekali tekan, sistem akan merespon dan menjalankan perintah sesuai dengan tombol yang ditekan dan akan berhenti sesuai batas yang ditentukan dalam program.

Tabel 6.2 merupakan hasil pengujian pengiriman perintah kendali lengan *gripper*. Pengiriman perintah dilakukan dari perangkat Android ke sistem mikrokontroller melalui sambungan *wifi*.

Tabel 6.2 Hasil Pengujian Pengiriman Perintah Kendali Lengan *Gripper*

No	Perintah Tombol	Pengujian ke-	Respon Sistem	Hasil
1	UP	1	Diam	Gagal
		2	Lengan *gripper* naik	Berhasil
		3	Lengan *gripper* naik	Berhasil
		4	Lengan *gripper* naik	Berhasil
		5	Lengan *gripper* naik	Berhasil
2	DOWN	1	Diam	Gagal
		2	Lengan *gripper* turun	Berhasil
		3	Lengan *gripper* turun	Berhasil
		4	Lengan *gripper* naik	Berhasil
		5	Lengan *gripper* turun	Berhasil

Pada tabel 6.2 dapat dilihat bahwa ada dua perintah yang digunakan yakni tombol "UP" sebagai menaikkan lengan gripper dan "DOWN" sebagai menurunkan lengan *gripper*. Dalam sekali tekan, sistem akan merespon dan menjalankan perintah sesuai dengan tombol yang ditekan dan akan berhenti sesuai batas *threshold* yang ditentukan dalam program.

6.1.4 Analisis Pengujian

Dari hasil pengujian diatas dapat disimpulkan bahwa pengiriman perintah dari aplikasi Android ke mikrokontroller NodeMCU mempunyai tingkat keberhasilan 100% untuk tombol ">> <<", 100% untuk tombol "<< >>", 80% untuk tombol "UP", dan 80% untuk tombol "DOWN".

6.2 Pengujian Akuisisi Data Nilai Sensor

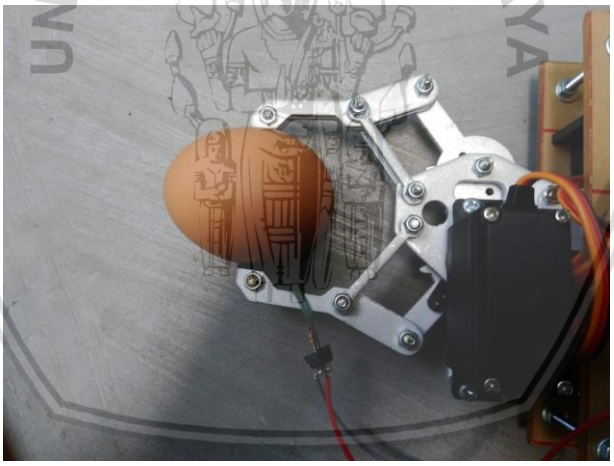
6.2.1 Tujuan Pengujian

Pengujian ini dilakukan untuk mengetahui keberhasil sistem mendapatkan nilai sensor FSR ketika sensor menyentuh objek dengan bantuan *gripper*. Data nilai sensor nantinya akan digunakan sebagai acuan ketika melewati *threshold.*

6.2.2 Prosedur Pengujian

Adapun prosedur pengujian akuisis data nilai sensor adalah sebagai berikut:

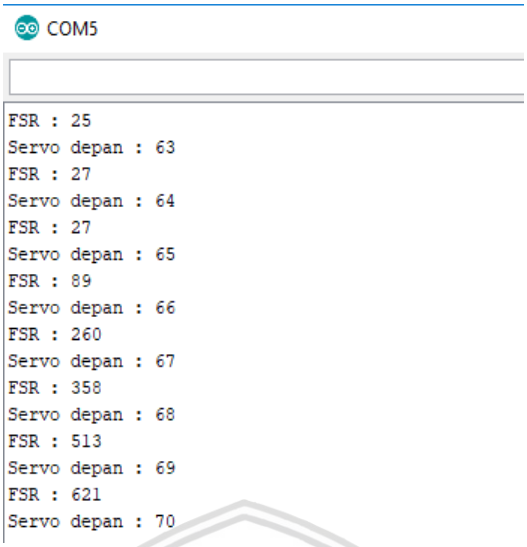
1. Masuk pada aplikasi Android.
2. Memasukkan *IP Address NodeMCU* pada tampilan aplikasi perangkat Android.
3. Menekan tombol sekali ”>> <<” atau “<< >>” agar *griper* bergerak menutup atau membuka.
4. Nilai sensor akan terbaca ketika sensor sudah menyentuh objek telur ayam seperti yang terlihat pada gambar 6.1



Gambar 6.1 Posisi Telur Ayam Ketika Dipegang Oleh *Gripper*

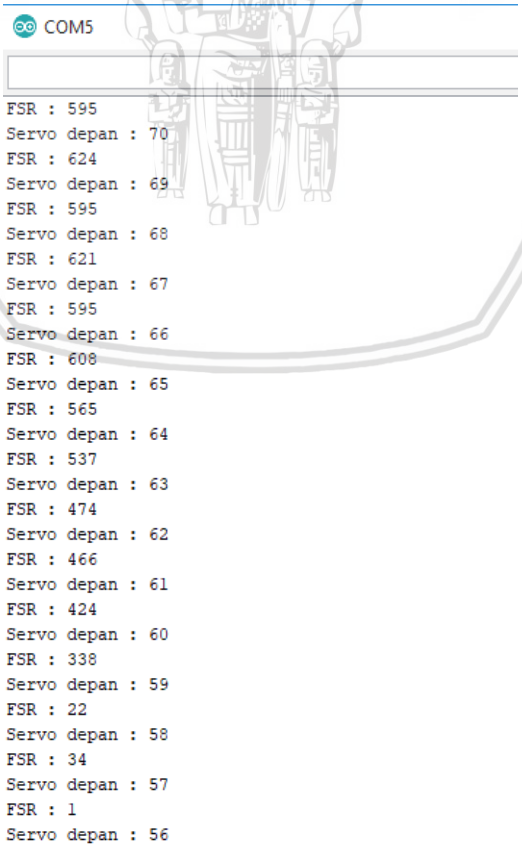
6.2.3 Hasil Pengujian

Gambar 6.2 menunjukkan bahwa ketika *gripper* bergerak untuk menyentuh objek, nilai sensor juga akan terbaca. Nilai sensor akan cenderung bertamabah dikarenakan *gripper* semakin menutup dengan bantuan motor servo. Nilai motor servo juga akan berubah dikarenakan bergerak menutup. Nilai sensor ditunjukkan pada “FSR” sedangkan nilai derajat posisi servo ditunjukkan pada “Servo depan”.



Gambar 6.2 Nilai Sensor ketika *Gripper* Menutup

Gambar 6.3 menunjukkan bahwa ketika *gripper* bergerak untuk menjauhi objek, nilai sensor juga akan terbaca. Nilai sensor akan cenderung berkurang dikarenakan *gripper* semakin terbuka dengan bantuan motor servo. Nilai motor servo juga akan berubah dikarenakan adanya gerakan membuka. Nilai sensor ditunjukkan pada "FSR" sedangkan nilai derajat posisi servo ditunjukkan pada "Servo depan".



Gambar 6.3 Nilai Sensor Ketika *Gripper* Terbuka

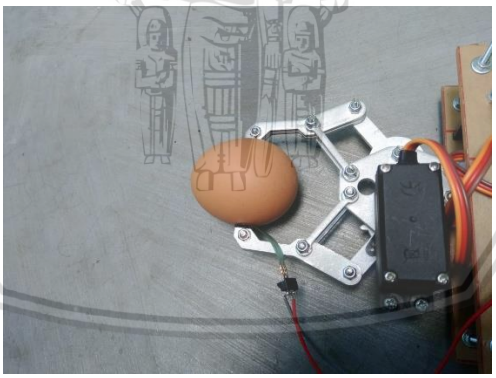
6.2.4 Analisis Pengujian

Dari hasil pengujian diatas dapat disimpulkan bahwa semakin erat *gripper* memegang objek telur ayam, maka semakin besar pula nilai sensor yang didapatkan. Sedangkan, semakin longgar *gripper* memegang objek telur ayam, maka semakin kecil pula nilai sensor yang didapatkan. Nilai sensor nantinya akan digunakan sebagai acuan kapan *gripper* harus berhenti, yakni setelah nilai sensor melewati *threshold*. Namun ketika *gripper* terbuka, pembacaan nilai sensor tetap dilakukan agar nilai *variable* sensor dapat berubah sesuai kondisi posisi sensor.

6.3 Pengujian *Gripper* Berhenti Otomatis terhadap *Threshold*

6.3.1 Tujuan Pengujian

Pengujian ini dilakukan untuk mengetahui keberhasilan *gripper* berhenti bergerak ketika nilai sensor sudah meewati nilai *threshold* yang telah ditentukan dalam program. Data sensor yang telah melewati nilai *threshold* akan dijadikan parameter keeratan pada objek yang digunakan sebagai bahan uji. Pemberhentian *gripper* dilakukan agar objek tidak rusak. Parameter keberhasilan pengujian dilihat dari objek yang tidak rusak namun tetap erat oleh *gripper*. Pengujian ini akan dilakukan sebanyak 10 kali. Contoh *gripper* memegang objek telur ayam seperti yang terlihat pada gambar 6.4.



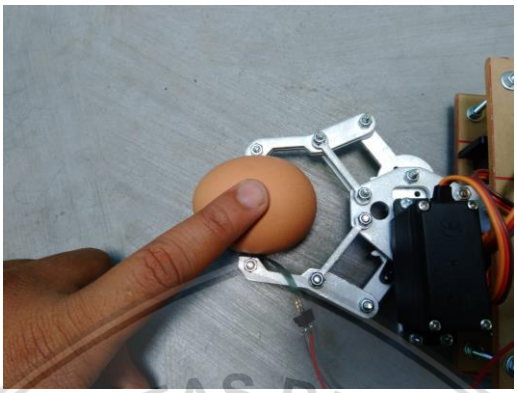
Gambar 6.4 *Gripper* Memegang Objek Telur Ayam

6.3.2 Prosedur Pengujian

Adapun prosedur pengujian *gripper* berhenti otomatis adalah sebagai berikut:

1. Masuk pada aplikasi Android.
2. Memasukkan *IP Address NodeMCU* pada tampilan aplikasi perangkat Android.
3. Memposisikan objek tepat ditengah *end effector gripper*.

4. Menekan tombol ”>> <<” sekali agar *gripper* menutup dan sensor dapat bersentuhan langsung dengan objek.
5. Memberi guncangan pada objek telur ayam untuk menguji apakah objek yang dipegang jatuh atau tidak, seperti gambar 6.5.



Gambar 6.5 Pengujian Keeratan Gripper dengan Guncangan

6.3.3 Hasil Pengujian

Tabel 6.3 merupakan hasil pengujian kendali *gripper* berhenti otomatis ketika nilai sensor sudah melewati threshold. Pengiriman perintah dilakukan dengan menekan tombol “>> <<” sekali pada aplikasi Android.

Tabel 6.3 Hasil Pengujian *Gripper* Berhenti Otomatis

Pengujian ke-	Pergerakan *Gripper*	Nilai Sensor	Gerakan Akhir *Gripper*	Posisi Objek	Hasil
1	Menutup	555	Berhenti	Tetap	Berhasil
2	Menutup	645	Berhenti	Tetap	Berhasil
3	Menutup	570	Berhenti	Tetap	Berhasil
4	Menutup	636	Berhenti	Tetap	Berhasil
5	Menutup	557	Berhenti	Tetap	Berhasil
6	Menutup	594	Berhenti	Tetap	Berhasil
7	Menutup	560	Berhenti	Tetap	Berhasil
8	Menutup	621	Berhenti	Tetap	Berhasil
9	Menutup	632	Berhenti	Tetap	Berhasil
10	Menutup	630	Berhenti	Tetap	Berhasil
Persentase Keberhasilan					100%

Pada tabel 6.3 dapat dilihat bahwa ada dua kondisi *gripper* yakni kondisi awal sebelum nilai sensor mencapai *threshold* sedangkan kondisi akhir setelah nilai sensor mencapai *threshold*. Dalam sekali tekan, sistem akan merespon dan

menjalankan perintah sesuai dengan tombol yang ditekan dan akan berhenti sesuai batas yang ditentukan dalam program.

6.3.4 Analisis Pengujian

Dari hasil pengujian diatas dapat disimpulkan bahwa pemberhentian *gripper* dengan menggunakan *threshold* dapat mencegah kerusakan yang ditimbulkan *gripper* ketika menjapit objek telur ayam. Pada objek yang sama, yaitu telur ayam diatur *threshold* 552. Dari hasil uji didapat bahwa ketika nilai sensor melewati *threshold*, *gripper* dapat berhenti dan objek tetap utuh. Persentase keberhasilan menjaga keutuhan objek telur ayam adalah 100%. Perbedaan nilai sensor yang didapat bisa jadi dikarenakan lapisan objek telur ayam yang berbeda tiap jenisnya atau tingkat sensitif sensor tiap waktu yang berubah-ubah.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan pengujian yang telah dilakukan, implementasi gripper pada end effector robot untuk memegang telur ayam dengan sensor FSR (Force Sensitive Resistor) dapat disimppulkan sebagai berikut:

1. Pengujian terhadap tekanan yang terdapat diantara end effector gripper dan objek telur ayam dilakukan dengan menggunakan sensor FSR (Force Sensitive Resistor). Berdasarkan pengujian yang telah dilakukan, didapat hasil bahwa sensor FSR akan semakin bertambah ketika sensor menyentuh objek semakin erat. Nilai sensor akan menurun ketika sensor mulai lepas dari objek.
2. Untuk menjaga keutuhan objek telur ayam, digunakanlah threshold sebagai alat bantu memberhentika gripper yang sedang bergerak menutup. Berdasarkan pengujian yang telah dilakukan, tingkat keberhasilan sistem memberhentikan gripper dan membuat objek elur ayam tetap utuh adalah 100%.
3. Untuk implementasi kontrol jarak jauh atau secara nirkabel digunakan koneksi wifi untuk menyambungkan client dan server. Dengan bantuan wifi, server dapat dengan mudah diketahui IP Address sebagai identitas server tersebut. Ada 4 tombol yang diuji, antara lain tombol GRIP, UNGRIP, UP, DOWN. Berdasarkan hasil pengujian, tombol GRIP dan UNGRIP mempunyai tingkat keberhasilan 100%, sedangkan tombol UP dan DOWN mempunyai tingkat keberhasilan 80%.

7.2 Saran

Adapun saran yang diajukan penulis agar dikemudian hari sistem dapat dikembangkan lebih lanjut adalah sebagai berikut:

1. Nilai *threshold* masih manual, maka dari itu perlu ditambahkan sistem agar dapat secara otomatis mengetahui *threshold* pada objek yang dipegang.
2. Servo masih kurang respon, perlu modifikasi agar servo cepat merespon.
3. Koneksi dengan wifi dirasa masih kurang sederhana, perlu adanya pengontrol secara nirkabel namun yang lebih sederhana.

DAFTAR PUSTAKA

Abelson, Hal. 2017. *Mengenal MIT App Inventor*. [online] MIT App Inventor. Tersedia di: <http://appinventor.mit.edu/explore/about-us.html> [Diakses 10 Juni 2018].

Aidan. 2016. *Our Arduino IDE Tutorial.* [online] core electronics. Tersedia di: <https://core-electronics.com.au/tutorials/arduino-ide-tutorial.html> [Diakses 10 Juni 2018].

Admin. 2017. *Apa itu Module NodeMCU ESP8266?.* [online] nyebarilmu. Tersedia di: <https://www.nyebarilmu.com/apa-itu-module-nodemcu-esp8266/> [Diakses 12 Juni 2018].

Amiroh. 2015. *Bekerja pada Blocks Viewer App Inventor.* [online] App Inventor. Tersedia di: <http://amiroh.web.id/bekerja-pada-blocks-viewer-app-inventor/> [Diakses 11 Juni 2018].

Astriani, Dwiarum. 2013. *Perbedaan http dan https.* [online] ilmukomputer. Tersedia di: <http://ilmukomputer.org/2013/01/30/perbedaan-http-dan-https/> [Diakses 13 Juni 2018].

Boga, Yasa. 2006. Resep Praktis dan Lezat, Telur: Padat Nutrisi, Ekonomis, Yummy. Jakarta: PT Gramedia Pustaka Utama.

Bourely, Antoine J., et al. 1987. *Robotics Egg Candling System.* California Agriculture.

Fuster, Anna Maria Gil. 2015. *Gripper Design and Development for A Modular Robot.* Denmark: Technical University of Denmark.

Gultekin, Hakan et al. 2017. *Pure cycles in two-machine dual-gripper robotic cells.* Robotics and Computer-Integrated Manufacturing.

Isworo, Hajar et al. 2015. *Analisis Tegangan Pada Gripper Pencekam Botol Menggunakan Simulasi.* Banjarmasin: Jurnal Sains dan Terapan Politeknik Hasnur. Vol.3, No.2.

Karim, Rahmatia. (2016). *Pentingnya Penggunaan Wi-Fi dalam Memenuhi Kebutuhan Informasi Pemustaka pada Kantor Perpustakaan dan Kearsipan Daerah Kota Tidore.* Journal Acta Diurna.

Kumar, Rahul, et al. 2016. *A Low Cost Linear Force Feedback Control System for a Two-Fingered Parallel Configuration Gripper.* IEEE International Symposium on Robotics and Intelligent Sensors.

Ladyada. 2013. *Force Sensitive Resistor (FSR).* Adafruit Industries.

Oyeyinka, Samson, et al. 2017. *Potentials of indigenous chicken eggs in the preparation of cake and chin chin.* Journal of the Saudi Society of Agricultural Sciences.

Pratama, F. Adi. 2017. *Mengenal Manfaat dan Kandungan dari Telur Ayam.* [online] Labsatu News. Tersedia di: <https://news.labsatu.com/mengenal-manfaat-dan-kandungan-dari-telur-ayam/> [Diakses 18 Juni 2018].

Purnama, Agus. 2012. *LCD (Liquid Crystal Display).* [online] Elektronika Dasar. Tersedia di: < http://elektronika-dasar.web.id/lcd-liquid-cristal-display/> [Diakses 16 Juni 2018].

Purnama, Agus. 2012. *Motor Servo.* [online] Elektronika Dasar. Tersedia di: <http://elektronika-dasar.web.id/motor-servo/> [Diakses 16 Juni 2018].

Susa'at, Sodikin. 2015. *Pengaturan Motor Robot Lengan (Arm Robotic) dengan Setting Point Gerakan Lengan Manusia Berbasis PID Menggunakan Mikrokontroller ATMEGA16.* [online] VEDC Malang. Tersedia di: <http://www.vedcmalang.com/pppptkboemlg/index.php/menuutama/listrik-electro/1321-pengaturan-motor-robot-lengan-arm-robotic-dengan-setting-point-gerakan-lengan-manusia-berbasis-pid-menggunakan-mikrokontroler-atmega16> [Diakses 12 Juni 2018].

Tomi. 2014. *I2C to 16x2 LCD Converter.* [online] raspberrypi. Tersedia di: <https://www.raspberrypi.org/forums/viewtopic.php?t=94820> [Diakses 19 Juni 2018].

Zarkasih, M. Eraz, et al. 2017. *Pengembangan Sistem Haptic untuk Memegang Objek (Gripper) dengan Komunikasi Wifi pada Mobile Robot.* Malang: Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer. Vol.2, No.6:2309-2318.